PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification [7] :
G06T 11/20

A1

(11) International Publication Number: WO 00/17819

(43) International Publication Date: 30 March 2000 (30.03.00)

(21) International Application Number: PCT/US99/22285

(22) International Filing Date: 24 September 1999 (24.09.99)

(30) Priority Data:

| | | |
|---|---|---|
| 60/101,927 | 24 September 1998 (24.09.98) | US |
| 60/102,523 | 30 September 1998 (30.09.98) | US |


(71)(72) Applicant and Inventor: ANANYA, Brigit [US/US]; 20725 Locust Drive, Los Gatos, CA 95033 (US).

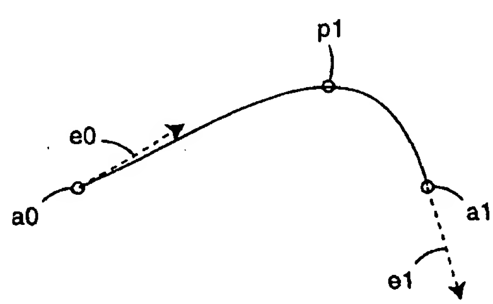
(74) Agents: POLLOCK, Michael, J. et al.; Limbach & Limbach L.L.P., 2001 Ferry Building, San Francisco, CA 94121 (US).


(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

**Published**

*With international search report.*
*Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.*


(54) Title: COMPUTER CURVE CONSTRUCTION SYSTEM AND METHOD




(57) Abstract

A computer curve construction system and method for generating curves is shown. Each curve consists of several curve components, connected either with $G^0$ continuity, i.e continuity of points, $G^1$ continuity, i.e. continuity of points and tangent directions, or $G^2$ continuity, i.e. continuity of points, tangent directions, and curvatures. There are different types of curve components, in a first embodiment, for a peak–point curve, the features are start and end points, start and end tangent directions, and a peak point that defines the greatest distance between the curve and the chord, i.e. the connecting line segment between the start and end point. During or after constructing a curve it is possible to modify the curve by changing the position of a feature of any curve component. However there is one exception: if the curve component is connected with $G^2$ continuity at the start or end point, the peak point cannot be changed.

*FOR THE PURPOSES OF INFORMATION ONLY*

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| AL | Albania | ES | Spain | LS | Lesotho | SI | Slovenia |
| AM | Armenia | FI | Finland | LT | Lithuania | SK | Slovakia |
| AT | Austria | FR | France | LU | Luxembourg | SN | Senegal |
| AU | Australia | GA | Gabon | LV | Latvia | SZ | Swaziland |
| AZ | Azerbaijan | GB | United Kingdom | MC | Monaco | TD | Chad |
| BA | Bosnia and Herzegovina | GE | Georgia | MD | Republic of Moldova | TG | Togo |
| BB | Barbados | GH | Ghana | MG | Madagascar | TJ | Tajikistan |
| BE | Belgium | GN | Guinea | MK | The former Yugoslav Republic of Macedonia | TM | Turkmenistan |
| BF | Burkina Faso | GR | Greece | | | TR | Turkey |
| BG | Bulgaria | HU | Hungary | ML | Mali | TT | Trinidad and Tobago |
| BJ | Benin | IE | Ireland | MN | Mongolia | UA | Ukraine |
| BR | Brazil | IL | Israel | MR | Mauritania | UG | Uganda |
| BY | Belarus | IS | Iceland | MW | Malawi | US | United States of America |
| CA | Canada | IT | Italy | MX | Mexico | UZ | Uzbekistan |
| CF | Central African Republic | JP | Japan | NE | Niger | VN | Viet Nam |
| CG | Congo | KE | Kenya | NL | Netherlands | YU | Yugoslavia |
| CH | Switzerland | KG | Kyrgyzstan | NO | Norway | ZW | Zimbabwe |
| CI | Côte d'Ivoire | KP | Democratic People's Republic of Korea | NZ | New Zealand | | |
| CM | Cameroon | | | PL | Poland | | |
| CN | China | KR | Republic of Korea | PT | Portugal | | |
| CU | Cuba | KZ | Kazakstan | RO | Romania | | |
| CZ | Czech Republic | LC | Saint Lucia | RU | Russian Federation | | |
| DE | Germany | LI | Liechtenstein | SD | Sudan | | |
| DK | Denmark | LK | Sri Lanka | SE | Sweden | | |
| EE | Estonia | LR | Liberia | SG | Singapore | | |

TITLE OF THE INVENTION

# COMPUTER CURVE CONSTRUCTION SYSTEM AND METHOD

## RELATED APPLICATIONS

This application claims the benefit of U.S. Provisional Application No. 60/101,927, filed September 24, 1998, by Brigit Ananya for Computer Curve Construction System, and of U.S. Provisional Application No. 60/102,523, filed September 30, 1998, by Brigit Ananya for Computer Curve Construction System.

## BACKGROUND OF THE INVENTION

Field Of The Invention

This invention relates generally to computer curve construction systems and methods.

Description of Related Art

A cubic Bezier curve is a mathematically defined curve. In a typical computer drawing program, it is drawn by setting four control points, i.e., by setting a start point $a_0$, a start tangent vector $v_0$, an end point $a_1$, and an end tangent vector $v_1$, as shown in Fig. 1. Additional Bezier curves (not shown) may be drawn, each connected to the end point of the previous Bezier curve, to create composite curves by setting additional points and tangent vectors. The length and direction of tangent vectors

$v_0$ and $v_1$ are defined by setting their corresponding end points. The depth of the curve is determined by the lengths of tangent vectors $v_0$ and $v_1$, and the slope of the curve is determined by the angles of tangent vectors $v_0$ and $v_1$. Longer tangent vectors produce a curve with greater depth, and more angled tangent vectors produce a curve with greater slope. If at least one tangent vector is long enough, such as tangent vector $v_1'$ in Fig. 2, the beginning of the curve is positioned on the other side of tangent vector $v_0$, and an inflection point b is created at an unpredictable position. The curve is curved clockwise to the left of inflection point b, and counterclockwise to the right of inflection point b. Using the lengths of tangent vectors to determine the depth of a curve is unintuitive and unpredictable, so that the shape of the curve is difficult to control.

With a typical computer drawing program, to be able to trace a curve such as the curve shown in Fig. 4 it usually takes several iterations. Fig. 5 shows a first and second attempt (curves $c_1$ and $c_2$) to trace the curve trying to adjust the length of the end tangent vector, which never achieves the desired curve. Fig. 6 shows a third and fourth attempt (curves $c_3$ and $c_4$) first adjusting the length of the start tangent vector instead of the end tangent vector, and then again adjusting the length of the end tangent vector. The end result is still not quite satisfactory (the peak-point, as described later, is not in the right place, it is too much to the left). Some computer drawing programs allow adjusting a point of the curve at any particular parameter. With this kind of adjustment it is easier to achieve the desired curve than with adjusting the length of the

tangent vectors. But it still takes several attempts, and it is not very intuitive, since the point does not have any geometric meaning (even if the original point selected is the peak point, as described later, it does not remain being the peak point, when it is adjusted). However, it is possible to trace the curve in one attempt by using the peak-point curve, which is described later.

In typical computer drawing program, when several Bezier curves are drawn as curve components of one composite curve, they sometimes are not connected very smooth such as the curve components $c_1$ and $c_2$ shown in Fig. 3, because they only connect with the continuity of points and tangent vectors. However, for curvature curves, which are described later, the curve components are connected very smooth such as the curve components $c_1$ and $c_2$ shown in Fig. 35, because they connect with the continuity of points, tangent directions and curvatures.

## SUMMARY OF THE INVENTION

An object of the present computer curve construction system is to enable the construction of curves more intuitively, predictably, and accurately. Further objects of the present invention will become apparent from a consideration of the drawings and ensuing description.

The computer curve construction allows constructing curves which consist of several curve components, which are connected either with $G^0$ continuity, i.e. continuity of points (geometric continuity of order 0), $G^1$ continuity, i.e. continuity of points and tangent directions, just the

slopes of the tangents, not the lengths (geometric continuity of order 1), or $G^2$ continuity, i.e. continuity of points, tangent directions, and curvatures (geometric continuity of order 2).

For each curve component certain features are used for its construction. There are different types of curve components, which use different features for their construction. In a first embodiment, for a peak-point curve, the features are start and end points, start and end tangent directions, and a peak point that defines the greatest distance between the curve and the chord, i.e. the connecting line segment between the start and end point. In a second embodiment, for a point-point curve, the features are start and end points, start tangent direction, and a peak point. In a third embodiment, for a point-tangent curve, the features are start and end points, and start and end tangent directions. In a fourth embodiment, for a point curve, the features are start and end points, and a start tangent direction. In a fifth embodiment, for a curvature curve, the features are start and end points, start and end tangent directions, and start and end curvatures. In a sixth embodiment, for a circular arc, the features are start point, start tangent direction, and end point. In a seventh embodiment, for a straight line segment, the features are start point, start tangent direction, and end point.

During or after constructing a curve it is possible to modify the curve by changing the position of a feature of any curve component. In general (if the curve component is not a straight line segment) all features can be changed: the start and end points, the start and end tangent

directions, the start and end curvatures, and the peak point. This means that not only the features that were used for the construction of the curve component can be changed, but also the other features, which were set automatically when the curve component was drawn so that they can be changed later. However there is one exception: if the curve component is connected with $G^2$ continuity at the start or end point, the peak point cannot be changed, because when a feature is changed, the types of continuity by which the curve component is connected at the start and end points remain the same.

It is also possible to modify the type of continuity by which two curve components are connected, or to make two curves out of one curve, or one curve out of two curves, to delete and redraw any curve component, or to add or subtract curve components.

This computer curve construction system enables the construction of curves more intuitively, predictably, and accurately, and it is also faster than the computer curve construction systems in typical computer drawing programs.

A better understanding of the features and advantages of the present invention will be obtained by reference to the following detailed description and accompanying drawings which set forth illustrative embodiments in which the principles of the invention are utilized.

## BRIEF DESCRIPTION OF THE DRAWINGS

Fig.1 illustrates a prior art curve.

Fig. 2 illustrates another prior art curve.

Fig. 3 illustrates another prior art curve.

Fig. 4 illustrates a curve to be traced.

Fig. 5 illustrates the first and second attempt of prior art curves to trace the curve.

Fig. 6 illustrates the third and fourth attempts of prior art curves to trace the curve.

Fig. 7 is a first step in constructing a peak-point curve.

Fig. 8 is a second step in constructing a peak-point curve.

Fig. 9 is a third step in constructing a peak-point curve.

Fig. 10 is a fourth step in constructing a peak-point curve.

Fig. 11 is a fifth step in constructing a peak-point curve.

Fig. 12 is a first step in constructing a point-point curve.

Fig. 13 is a second step in constructing a point-point curve.

Fig. 14 is a third step in constructing a point-point curve.

Fig. 15 is a fourth step in constructing a point-point curve.

Fig. 16 is a fifth step in constructing a point-point curve.

Fig. 17 is a sixth step in constructing a point-point curve.

Fig. 18 is a seventh step in constructing a point-point curve.

Fig. 19 is a first step in constructing a point-tangent curve.

Fig. 20 is a second step in constructing a point-tangent curve.

Fig. 21 is a third step in constructing a point-tangent curve.

Fig. 22 is a fourth step in constructing a point-tangent curve.

Fig. 23 is a fifth step in constructing a point-tangent curve.

Fig. 24 is a first step in constructing a point curve.

Fig. 25 is a second step in constructing a point curve.

Fig. 26 is a third step in constructing a point curve.

Fig. 27 is a fourth step in constructing a point curve.

Fig. 28 is a fifth step in constructing a point curve.

Fig. 29 is a first step in constructing a curvature curve.

Fig. 30 is a second step in constructing a curvature curve.

Fig. 31 is a third step in constructing a curvature curve.

Fig. 32 is a fourth step in constructing a curvature curve.

Fig. 33 is a fifth step in constructing a curvature curve.

Fig. 34 is a sixth step in constructing a curvature curve.

Fig. 35 is a final illustration of the constructed curvature curve.

Fig. 36 is a first step in constructing a straight line.

Fig. 37 is a second step in constructing a straight line.

Fig. 38 is a third step in constructing a straight line.

Fig. 39 is a first step in constructing a circle.

Fig. 40 is a second step in constructing a circle.

Fig. 41 is a third step in constructing a circle.

Fig. 42 is a flow chart showing a method for constructing a peak-point, point-point, point-tangent, or point curve as the first component or as the nth component connected with $G^0$ continuity to the (n-1)th component. (Setting the start point $a_0$ is only for the first component.)

Fig. 43 is a flow chart showing a method for constructing a peak-point, point-point, point-tangent, or point curve as the nth component connected with $G^1$ continuity to the (n-1)th component.

Fig. 44 is a flow chart showing a method for constructing a curvature curve as the first component or the n'th component connected with $G^0$ continuity to the (n-1)th component. (Setting the start point $a_0$ is only for the first component.)

Fig. 45 is a flow chart showing a method for constructing a curvature curve as the n'th component connected with $G^1$ continuity to the (n-1)th component.

Fig. 46 is a flow chart showing a method for constructing a curvature curve as the n'th component connected with $G^2$ continuity to the (n-1)th component.

Fig. 47 is a view of one embodiment of the system for practicing the invention.

## DETAILED DESCRIPTION OF THE INVENTION

Introduction

The present computer curve construction system and method is preferably implemented as part of a computer drawing program for drawing curves, which is a practical application in the industrial art of computer drawing. In one embodiment, the system may be implemented in any suitable computer language for any operating system and any hardware platform. The system may be distributed through any medium, such as a disc, non-volatile memory, or being available for downloading on a network. In other embodiments the system may be implemented in any firmware or any hardware.

Fig. 47 shows a computer system on which the computer construction system can be implemented. The computer typically includes a CPU 1, a storage device 2 such as a hard disk drive, RAM 3, ROM 4, a clock 5, a video driver 6, and some peripherals such as a video monitor 11, input devices 12, and a plotter or printer 13. In one embodiment , the storage device stores a computer program, which is operative, with the processor, to perform the steps and methods discussed herein.

In Figs.1-41, all points are represented by small circles, and all tangent directions are represented by dashed arrows. A tangent direction only indicates the direction of a tangent vector; it does not indicate the length of the tangent vector. A point may be set by moving a cursor to a position and pressing a button on an input device (such as a mouse). A point and its corresponding tangent direction may be set in any one of a variety of ways well known in the art, for example, the point may be set by moving a cursor to a first position and pressing a button on an input device (such as a mouse), and the tangent direction may be set by dragging the cursor to a second position and releasing the button. The direction from the first position to the second position is the tangent direction. Only the radial direction from the first position to the second position is important; the distance between them is irrelevant. As discussed herein the curve components can be constructed as Bezier curves or circular arcs. It should be noted, however, that a wide range of curve types and formulas could be used to construct the curve components.

A curve component is connected with $G^0$ continuity to the previous curve component if its start point equals the end point of the previous curve component, it is connected with $G^1$ continuity to the previous curve component if its start point and start tangent direction equal the end point and end tangent direction of the previous curve component, and it is connected with $G^2$ continuity to the previous curve component if its start point, start direction, and start curvature equal the end point, end tangent direction, and end curvature of the previous curve component. The previous curve component is said to be connected to the curve component with the same continuity as the curve component is connected to the previous curve component. Since two neighbor curve components are at least connected by $G^0$ continuity, they always connect at a point. These connecting points are called anchor points, and the curve components are said to be connected with different types of continuity at the anchor points.

Figs.7-11 — Peak-Point Curves

A first embodiment of the computer curve construction system is for constructing peak-point curves. In a first step shown in Fig. 7, a start point $a_0$ is set, and a start tangent direction $e_0$ is set. In a second step shown in Fig. 8, an end point $a_1$ is set, and an end tangent direction $e_1$ is set. In a third step shown in Fig. 9, a peak point $p_1$ is set between start point $a_0$ and end point $a_1$. A cubic Bezier curve $c_1$ is automatically drawn through points $a_0$, $p_1$, and $a_1$, with start and end tangent directions $e_0$ and $e_1$ and with peak point $p_1$, according to any suitable set of mathematical formulas.

The peak point $p_1$ is the point at the greatest distance $d_1$ between the curve $c_1$ and an imaginary line segment connecting the start point $a_0$ and the end point $a_1$. This imaginary line segment is often referred to as the chord. Since the peak point is the point at the greatest distance from the chord, the tangent vector at the peak point is parallel to the chord. The tangent direction at the peak point equals the chord vector, i.e. the vector from the start point to the end point. If the chord is horizontal, the peak point is the point at the peak.

This cubic Bezier curve $c_1$ is already drawn when the mouse button is pressed for the peak point $p_1$, and when the mouse is dragged, the peak point $p_1$ is dragged to a new position, and the cubic Bezier curve $c_1$ is changed, the final shape of which is drawn when the mouse is released.

Additional curve components of any type may be constructed to connect with $G^2$, $G^1$, or $G^0$ continuity. In optional additional steps shown in Figs.10 and 11, a second peak-point curve $c_2$ connected to curve component $c_1$ with $G^1$ continuity is constructed by setting an end point $a_2$, an end tangent direction $e_2$, and a peak point $p_2$. A cubic Bezier curve $c_2$ is automatically drawn through points $a_1$, $p_2$, and $a_2$, with start and end tangent directions $e_1$ and $e_2$ and peak point $p_2$. A composite curve is thus created with two curve components. If the second peak-point curve is connected with $G^0$ continuity, a new start tangent direction $e_1'$ has to be set first before setting an end point $a_2$, an end tangent direction $e_2$, and a peak point $p_2$. A cubic Bezier curve $c_2$ is

automatically drawn through points $a_1$, $p_2$, and $a_2$, with start and end tangent directions $e_1'$ and $e_2$ and peak point $p_2$.

The peak-point curves are more intuitive, predictable, and accurate to construct than curve components with typical computer drawing programs, because the peak point directly determines its path.

An exemplar set of formulas for determining the cubic Bezier curve of a peak-point curve is as follows:

The equation for the cubic Bezier curve is

$$x(t) = (1-t)^3 b_0 + 3t(1-t)^2 b_1 + 3t^2(1-t) b_2 + t^3 b_3,$$

where $b_0$, $b_1$, $b_2$, $b_3$ are the control points. The first derivative of the cubic Bezier curve is

$$x'(t) = 3((1-t^2)(b_1 - b_0) + 2t(1-t)(b_2 - b_1) + t^2(b_3 - b_2)).$$

The start and end points are $a_0$ and $a_1$. The start and end tangent vectors are $x'(0) = \lambda e_0$ and $x'(1) = \mu e_1$, where $\lambda$ and $\mu$ are positive real numbers and $e_0$ and $e_1$ are vectors of length 1. Again, the peak point $p_1$ is the point on the curve at the greatest distance from the chord. Let $\tau$ be the parameter for $p_1$.

Then

$$\begin{aligned} b_0 &= a_0 \\ b_1 &= \frac{\lambda}{3} e_0 + a_0 \\ b_2 &= a_1 - \frac{\mu}{3} e_1 \\ b_3 &= a_1. \end{aligned}$$

There are 2 cases with the following assumptions, where "sign" means signature and "×" denotes the vector product between two 2-dimensional vectors, which equals the determinant of the matrix formed by the two vectors:

1) If $e_0 \times e_1 \neq 0$, assume that $\mathrm{sign}(e_0 \times (a_1 - a_0)) = \mathrm{sign}(e_1 \times (a_0 - a_1)) \neq 0$ and that the peak point $p_1$ lies inside the area of the ray corresponding to the start tangent direction, the ray opposite to the end tangent direction, and the chord.

2) If $e_0 \times e_1 = 0$, also assume that $\mathrm{sign}(e_0 \times (a_1 - a_0)) = \mathrm{sign}(e_1 \times (a_0 - a_1)) \neq 0$ and that the peak point $p_1$ lies inside the area of the ray corresponding to the start tangent direction, the ray opposite to the end tangent direction, and the chord.

Let it be mentioned that for peak-point curves, and also for point-point, point-tangent, point, and curvature curves, the assumption that $\mathrm{sign}(e_0 \times (a_1 - a_0)) = \mathrm{sign}(e_1 \times (a_0 - a_1))$ is only for a particular embodiment for the purpose of avoiding that the Bezier curve has an inflection point, and the assumption is only a sufficient assumption not a necessary assumption. Also, the additional assumption that $\mathrm{sign}(e_0 \times (a_1 - a_0)) \neq 0$ is only for a particular embodiment for the purpose of avoiding that the cubic Bezier curve is a straight line segment so that the straight line segment can be constructed explicitly.

The following formulas are provided for these two cases above:

1) For $\tau$ there is the following algebraic equation of order 5:

$$f(\tau) = (e_1 \times (a_0 - s_1))\,(e_0 \times (a_1 - a_0))\,(\tau - 4\tau^2 + 3\tau^3)$$
$$+ (e_0 \times (a_0 - s_1))\,(e_1 \times (a_0 - a_1))\,(2\tau - 5\tau^2 + 3\tau^3)$$
$$+ (e_0 \times (a_1 - a_0))\,(e_1 \times (a_0 - a_1))\,(3\tau^3 - 5\tau^4 + 2\tau^5) = 0,$$

and for $\lambda$ and $\mu$

$$= \frac{1}{(e_0 \times e_1)\tau(1-\tau)^2}(e_1 \times (a_0 - s_1) + (e_1 \times (a_0 - a_1))(-3\tau^2 + \tau^3))$$
$$= \frac{1}{(e_0 \times e_1)\tau^2(1-\tau)}(e_0 \times (a_0 - s_1) + (e_0 \times (a_1 - a_0))(3\tau^2 - 2\tau^3))$$

2) For $\tau$ there is the following algebraic equation of order 3:

$$f(\tau) = e_1 \times (a_0 - s_1) + (e_1 \times (a_0 - a_1))\,(-3\tau^2 + 2\tau^3) = 0,$$

and for $\lambda$ and $\mu$ the following system of linear equations

if $a_0 \neq 0$

$$(e_0 \times a_0)\,(\tau\,(1 - \tau^2))\,\lambda - (e_1 \times a_0)\,(\tau^2\,(1 - \tau))\,\mu$$
$$- s_1 \times a_0 + (a_1 \times a_0)\,(3\tau^2 - 2\tau^3) = 0$$

or if $a_0 = 0$

$$(e_0 \times a_1)\,(\tau\,(1 - \tau^2))\,\lambda - (e_1 \times a_1)\,(\tau^2\,(1 - \tau))\,\mu$$
$$- s_1 \times a_1 + (a_0 \times a_1)\,(1 - 3\tau^2 + 2\tau^3) = 0$$

and

$$(e_0 \times (a_1 - a_0))\,(1 - 4\tau + 3\tau^2)\,\lambda + (e_1 \times (a_0 - a_1))\,(2\tau - 3\tau^2)\,\mu = 0.$$

For the two algebraic equations there are the following solutions: The two algebraic equations are solved the same way. Let the derivative $f'(\tau)$ be unequal to 0 at the iterated solutions below. As an initial solution

$$\tau_0 = 0.5$$

is taken, and the algebraic equations are solved through Newton's iteration method

$$\tau_{i+1} = \tau_i - \frac{f(\tau_i)}{f'(\tau_i)},$$

where for case 1) above

$$f'(\tau_i) = (e_1 \times (a_0 - s_1))\,(e_0 \times (a_1 - a_0))\,(1 - 8\tau_i + 9\tau_i^2)$$
$$+ (e_0 \times (a_0 - s_1))\,(e_1 \times (a_0 - a_1))\,(2 - 10\tau_i + 9\tau_i^2)$$
$$(e_0 \times (a_1 - a_0))\,(e_1 \times (a_0 - a_1))\,(9\tau_i^2 - 20\tau_i^3 + 10\tau_i^4)$$

and for case 2) above

$$f'(\tau_i) = e_1 \times (a_0 - a_1)\,(-6\tau_i + 6\tau_i^2).$$

If the peak point lies close to any of the rays corresponding to the start tangent direction or opposite of the end tangent direction, the algebraic equations have no solution, and the formulas for the following two point-tangent curves (which are connected with $G^1$ continuity), as described later, are used: For the first point-tangent curve the start point and start tangent direction are the start point and start tangent direction of the peak-point curve, and the end point and end tangent direction are the peak point of the peak-point curve and the tangent direction at the peak point. For the second point-tangent curve the start point and start tangent direction are the peak point of the peak point curve and the tangent direction at the peak point, and the end point and end tangent direction are the end point and end tangent direction of the peak-point curve.

Figs.12-18 — Point-Point Curves

A second embodiment of the computer curve construction system is for constructing point-point curves. In a first step shown in Fig. 12, a start point $a_0$ is set, and a start tangent direction $e_0$ is set. In a second step shown in Fig. 13, a peak point $p_1$ is set. In a third step shown in Fig. 14, an end point $a_1$ is set. A cubic Bezier curve $c_1$ is automatically drawn through points $a_0$, $p_1$, and $a_1$, with start tangent direction $e_0$ and peak point p1, according to any suitable set of mathematical formulas. As shown in Fig. 15, an end tangent direction $e_1$ is automatically set in a symmetric way such that the angle between the chord vector and the end tangent direction equals the angle between the start tangent direction and the chord vector. (However it could be set in any other way as well.)

This cubic Bezier curve $c_1$ is already drawn when the mouse button is pressed for the end point $a_1$, and when the mouse is dragged, the end point $a_1$ is dragged to a new position, and the cubic Bezier curve $c_1$ is changed, the final shape of which is drawn when the mouse is released.

Additional curve components of any type may be constructed to connect with $G^2$, $G^1$, or $G^0$ continuity. In optional additional steps shown in Figs.16 and 17, a second point-point curve $c_2$ connected to curve component $c_1$ with $G^1$ continuity is constructed by setting a peak point $p_2$ and an end point $a_2$. A cubic Bezier curve $c_2$ is automatically drawn through points $a_1$, $p_2$ and $a_2$, with start tangent direction $e_1$ and peak point $p_2$. As shown in Fig. 18, an end tangent direction $e_2$ is

automatically set in the same way as for the first curve component $c_1$. A composite curve is thus created with two curve components. If the second point-point curve is connected with $G^0$ continuity, a new start tangent direction $e_1'$ has to be set first before setting a peak point $p_2$ and an end point $a_2$. A cubic Bezier curve $c_2$ is automatically drawn through points $a_1$, $p_2$ and $a_2$, with start tangent direction $e_1'$ and peak point $p_2$. Again an end tangent direction $e_2$ is automatically set in the same way as for the first curve component curve $c_1$.

The point-point curves are easy to construct, because the end tangent direction of each curve component is automatically determined.

An exemplar set of formulas for determining the cubic Bezier curve of a point-point curve is as follows:

The following assumption is made:
Assume that $\mathrm{sign}(e_0 \times (a_1 - a_0)) \neq 0$, and assume that the peak point $p_1$ lies inside the area of the ray corresponding to the start tangent direction, the ray opposite to the end tangent direction, and the chord.

After determining the end tangent direction $e_1$ as described above, the formulas for a point-point curve are the same as the formulas for a peak-point curve.

<u>Figs.19-23 — Point-Tangent Curves</u>

A third embodiment of the computer curve construction system is for constructing point-tangent curves. In a first step shown in Fig. 19, a

start point $a_0$ and a start tangent direction $e_0$ are set. In a second step shown in Fig. 20, an end point $a_1$ and an end tangent direction $e_1$ are set. A quadratic Bezier curve $c_1$ is automatically drawn through start and end points $a_0$ and $a_1$, with start and end tangent directions $e_0$ and $e_1$, according to any suitable set of mathematical formulas. As shown in Fig. 21, a peak point $p_1$ is automatically set by computing the peak point of the quadratic Bezier curve $c_1$. (However it could be set in any other way as well and a cubic Bezier curve could be drawn with the set peak point).

This quadratic Bezier curve $c_1$ is already drawn when the mouse is dragged to the position of the end tangent direction $e_1$, and when the mouse is dragged further, the end tangent direction $e_1$ is dragged to a new position, and the quadratic Bezier curve $c_1$ is changed, the final shape of which is drawn when the mouse is released.

Additional curve components of any type may be constructed to connect with $G^2$, $G^1$, or $G^0$ continuity. In an optional additional step shown in Fig. 22, a second point-tangent curve $c_2$ connected to curve component $c_1$ with $G^1$ continuity is constructed by setting an end point $a_2$ and an end tangent direction $e_2$. A second quadratic Bezier curve $c_2$ is automatically drawn through start and end points $a_1$ and $a_2$, with start and end tangent directions $e_1$ and $e_2$. As shown in Fig. 23, a peak point $p_2$ is automatically set by computing the peak point of the quadratic Bezier curve $c_2$. A composite curve is thus created with two curve components. If the point-tangent curve is connected with $G^0$ continuity, a

new start tangent direction $e_1'$ has to be set first before setting end point $a_2$ and an end tangent direction $e_2$. A quadratic Bezier curve $c_2$ is automatically drawn through points $a_1$ and $a_2$, with start and end tangent directions $e_1'$ and $e_2$. Again a peak point $p_2$ is automatically set by computing the peak point of the quadratic Bezier curve $c_2$.

The point-tangent curves are easy to construct, because the peak point of each curve component is automatically determined.

An exemplar set of formulas for determining the quadratic Bezier curve of a point-tangent curve is as follows:

The equation for the quadratic Bezier curve is

$$x(t) = (1-t)^2 b_0 + 2t(1-t)b_1 + t^2 b_2,$$

where $b_0$, $b_1$, $b_2$ are the control points. The first derivative of the quadratic Bezier curve is

$$x'(t) = 2((1-t)(b_1 - b_0) + t(b_2 - b_1)).$$

The start and end points are $a_0$ and $a_1$. The start and end tangent vectors are $x'(0) = \lambda e_0$ and $x'(1) = \mu e_1$, where $\lambda$ and $\mu$ are positive real numbers and $e_0$ and $e_1$ are vectors of length 1.

Then

$$b_0 = a_0$$
$$b_1 = \frac{\lambda}{2}e_0 + a_0 = a_1 - \frac{\mu}{2}e_1$$
$$b_2 = a_1.$$

The following assumptions are made:

Assume that $\text{sign}(e_0 \times (a_1 - a_0)) = \text{sign}(e_1 \times (a_0 - a_1)) = \text{sign}(e_0 \times e_1) \neq 0$.

The following formulas are provided:

For $\lambda$ and $\mu$

$$\lambda = \frac{2e_1 \times (a_o - a_1)}{(e_0 \times e_1)}$$

$$\mu = \frac{2e_0 \times (a_1 - a_0)}{(e_0 \times e_1)}$$

and for the parameter $\tau$ of the peak point which has the greatest distance from the chord

$$\tau = 0.5$$

(The parameter for the peak point of a quadratic Bezier curve is always 0.5. However this is not true for cubic Bezier curves.)

Figs.24-28 — Point Curves

A fourth embodiment of the computer curve construction system is for constructing point curves. In a first step shown in Fig. 24, a start point $a_0$ and a start tangent direction $e_0$ are set. In a second step shown in Fig. 25, an end point $a_1$ is set. A quadratic Bezier curve $c_1$ is automatically drawn through start and end points $a_0$ and $a_1$ with start tangent direction $e_0$, to any suitable set of mathematical formulas. As shown in Fig. 26, an end tangent direction $e_1$ for end point $a_1$ is automatically set in a symmetric way such that equals the angle between the chord vector and the end tangent direction equals the angle between the start tangent direction and the chord, and a peak point $p_1$ is also automatically set by computing the peak point of the quadratic Bezier curve $c_1$. In this embodiment the whole curve

component $c_1$ is symmetric with respect to the axis that is perpendicular to the chord and goes through the center of the chord. (However both the end tangent direction and the peak point could be set in any other way as well and a cubic Bezier curve could be drawn with the set end tangent direction and set peak point).

This quadratic Bezier curve $c_1$ is already drawn when the mouse button is pressed for the end point $a_1$, and when the mouse is dragged, the end point $a_1$ is dragged to a new position, and the quadratic Bezier curve $c_1$ is changed, the final shape of which is drawn when the mouse is released.

Additional curve components of any type may be constructed to connect with $G^2$, $G^1$, or $G^0$ continuity. In an optional additional step shown in Fig. 27, a second point curve $c_2$ connected to curve component $c_1$ with $G^1$ continuity is constructed by simply setting an end point $a_2$. A second quadratic Bezier curve $c_2$ is automatically drawn through start and end points $a_1$ and $a_2$, with start tangent direction $e_1$. As shown in Fig. 28, an end tangent direction $e_2$ is automatically set in the same way as for the first curve component $c_1$, and a peak point $p_2$ is also automatically set by computing the peak point of the quadratic Bezier curve. A composite curve is thus created with two curve components. If the second point curve is connected with $G^0$ continuity, a new start tangent direction $e_1'$ has to be set first before setting the end point $a_2$. A second quadratic Bezier curve $c_2$ is automatically drawn through points $a_1$ and $a_2$, with start and end tangent directions $e_1'$ and $e_2$. Again an end tangent

direction $e_2$ is automatically set in a symmetric way such that the angle between the chord vector and the end tangent direction equals the angle between the start tangent direction and the chord vector, and a peak point $p_2$ is also automatically set by computing the peak point of the quadratic Bezier curve.

The point curves are easy to construct, because the end tangent direction and the peak point of each curve component are automatically determined.

An exemplar set of formulas for determining the quadratic Bezier curve of a point curve is as follows:

The following assumption is made:
Assume that the angle between $e_0$ and $a_1 - a_0$ is smaller than 90° and larger than 0°.

After determining the end tangent direction $e_1$ as described above, the formulas for a point curve are the same as the formulas for a point-tangent curve.

If the point curve is the first curve component or the nth curve component that is connected with $G^0$ continuity to the (n-1)th curve component, it can also be constructed in a differenent way. It can be constructed by setting (a start point $a_0$ if it is the first curve component

and) a peak point $p_n$ and an end point $a_n$. The formula for this construction is

$$b_1 = 2(p_n - 0.25a_{n-1} - 0.25a_n),$$

Where $b_1$ is the control point and $a_{n-1}$ is the end point of the (n-1)th component .

Figs.29-35 — Curvature Curves

The embodiment of the curvature curve construction system allows for constructing curvature curves. In a first step shown in Fig. 29, a start point $a_0$ and a start tangent direction $e_0$ are set. In a second step shown in Fig. 30, and end point $a_1$ and an end tangent $e_1$ are set. If the angle between the start tangent direction and the end tangent direction is smaller than 180°, the start and end curvatures have to either both be small or both be large depending on the position ot the start and end points and the start and end tangent directions. Once these features are drawn, the rays $l_0$ and $l_1$ perpendicular to the tangent directions on the same side of the tangent directions as the chord are drawn together with limit points $L_0$ and $L_1$ on the rays, which show that the centers of the curvature circles have to either both be before these limit points or after these limit points. In a third step shown in Fig. 31, a center $m_0$ of a curvature circle (osculating circle) $r_0$ for the start curvature is set on the ray $l_0$. Since placing a cursor (not shown) exactly on $l_0$ is difficult, center $m_0$ is projected onto $l_0$ automatically from the cursor. Once the start curvature is drawn, the line segment or ray of $l_0$ on the same side of the start limit point $L_0$ as $m_0$, which is called start curvature line $cl_0$, is drawn, and the line segment or ray of $l_1$ on the same side of the end limit point

$L_1$, which is called end curvature line $cl_1$, is drawn. (In the case that the angle between the start tangent direction and the end tangent direction is 180° or larger, there are no limit points, and $cl_0 = l_0$ and $cl_1 = l_1$.) In a fourth step shown in Fig. 32, a center $m_1$ of a curvature circle for the end curvature is set on $cl_1$. Since placing a cursor (not shown) exactly on $cl_1$ is difficult, center $m_1$ is projected onto $cl_1$ automatically from the cursor. If, as an example, a curvature of zero is desired for the start or end curvature, the center of the curvature circle would be at infinity, but the center is shown at the point and the cursor is placed anywhere on the line of the tangent direction at the point and gets projected onto the point. A cubic Bezier curve $c_1$ is automatically drawn through start and end points $a_0$ and $a_1$ with start and end tangent directions $e_0$ and $e_1$ and start and end curvature circle centers $m_0$ and $m_1$ according to any suitable set of mathematical formulas. The start curvature of the cubic Bezier curve $c_1$ is defined by the curvature of the start curvature circle $r_0$, and the end curvature of the cubic Bezier curve $c_1$ is defined by the curvature of the end curvature circle $r_1$.

Additional curve components of any type may be constructed to connect with $G^2$, $G^1$, or $G^0$ continuity. In an optional fifth and sixth step as shown in Figs.33 and 34 an end point $a_2$, an end tangent direction $e_2$ and a center $m_2$ of an end curvature circle of a second curvature curve $c_2$ connected to curve $c_1$ with $G^2$ continuity is constructed. Once the end point and the end tangent direction are constructed, the end curvature line $cl_2$ is drawn, and the new start curvature line $cl_1$ is drawn, which is the intersection of the start curvature line of the second curve

component with start limit point $L_1'$ and the end curvature line of the first curve component. A cubic Bezier curve $c_2$ is automatically drawn through start and end points $a_1$ and $a_2$, with start and end tangent directions $e_1$ and $e_2$, with the set start and end curvatures. The final curve consisting of curve component $c_1$ and curve component $c_2$ is shown in Fig.35. If the second curvature curve is connected with $G^1$ continuity, a new center $m_1'$ of the start curvature circle has to be set after setting an end point $a_2$ and an end tangent direction $e_2$, and before setting a center $m_2$ of an end curvature circle. A cubic Bezier curve $c_2$ is automatically drawn through start and end points $a_1$ and $a_2$, with start and end tangent directions $e_1$ and $e_2$, with the set start and end curvatures. If the second curvature curve is connected with $G^0$ continuity, a new start tangent $e_1'$ has to be set first, and a new center $m_1'$ of the start curvature circle has to be set after setting an end point $a_2$ and an end tangent direction $e_2$, and before setting a center $m_2$ of an end curvature circle. A cubic Bezier curve $c_2$ is automatically drawn through points $a_1$ and $a_2$, with start and end tangent directions $e_1$ and $e_2$, with the set start and end curvatures. The curvature curve is very smooth if $G^2$ continuity is used, because adjacent curves are connected not only with the same tangent direction, but also with the same curvature.

An exemplar set of formulas for determining the curvature curve is as follows:

The equation for the cubic Bezier curve is

$$x(t) = (1-t)^3 b_0 + 3t(1-t)^2 b_1 + 3t^2(1-t) b_2 + t^3 b_3,$$

where $b_0$, $b_1$, $b_2$, $b_3$ are the control points. The first derivative of the cubic Bezier curve is

$$x'(t) = 3((1 - t^2)(b_1 - b_0) + 2t(1 - t)(b_2 - b_1) + t^2(b_3 - b_2)).$$

The second derivative of the cubic Bezier curve is

$$x''(t) = 6((1 - t)(b_2 - 2b_1 + b_0) + t(b_3 - 2b_2 + b_1)).$$

The curvature k(t) at the parameter t is

$$k(t) = \frac{x'(t) \times x''(t)}{|x'(t)|^3},$$

where "×" denotes the vector product between two 2-dimensional vectors. The start and end points are $a_0$ and $a_1$. The start and end tangent vectors are $x'(0) = \lambda e_0$ and $x'(1) = \mu e_1$, where $\lambda$ and $\mu$ are positive real numbers and $e_0$ and $e_1$ are vectors of length 1. The start and end curvatures are $k_0$ and $k_1$.

Then

$$\begin{aligned} b_0 &= a_0 \\ b_1 &= \frac{\lambda}{3} e_0 + a_0 \\ b_2 &= a_1 - \frac{\mu}{3} e_1 \\ b_3 &= a_1. \end{aligned}$$

If the curve is a straight line, then $e_0 = e_1$, and

$$\lambda = \mu = |a_1 - a_0|,$$

which implies $x(t) = a_0 + t(a_1 - a_0)$. If the curve is not straight, there are 5 cases with the following assumptions:

1) If $k_0 = 0$ and $k_1 = 0$, assume that $\operatorname{sign}(e_0 \times (a_1 - a_0)) = \operatorname{sign}(e_1 \times (a_0 - a_1)) = \operatorname{sign}(e_0 \times e_1) \neq 0$.

2) If $k_0 \neq 0$ and $k_1 = 0$, assume that $\operatorname{sign}(k_0) = \operatorname{sign}(e_0 \times (a_1 - a_0)) = \operatorname{sign}(e_1 \times (a_0 - a_1)) = \operatorname{sign}(e_0 \times e_1)$, and assume that

$$|k_0| < \frac{2(e_0 \times e_1)^2 |e_0 \times (a_1 - a_0)|}{3(e_1 \times (a_0 - a_1))^2}$$

3) If $k_0 = 0$ and $k_1 \neq 0$, assume that $\operatorname{sign}(k_1) = \operatorname{sign}(e0 \times (a_1 - a_0)) = \operatorname{sign}(e_1 \times (a_0 - a_1)) = \operatorname{sign}(e_0 \times e_1)$, and assume that

$$|k_1| < \frac{2(e_0 \times e_1)^2 |e_1 \times (a_0 - a_1)|}{3(e_0 \times (a_1 - a_0))^2}$$

4) If $k_0 \neq 0$ and $k_1 \neq 0$, and $e_0 \times e_1 = 0$, assume that $\operatorname{sign}(k_0) = \operatorname{sign}(k_1) = \operatorname{sign}(e_0 \times (a_1 - a_0)) = \operatorname{sign}(e_1 \times (a_0 - a_1))$.

5) If $k_0 \neq 0$ and $k_1 \neq 0$, and $e_0 \times e_1 \neq 0$, assume that $\operatorname{sign}(k_0) = \operatorname{sign}(k_1) = \operatorname{sign}(e_0 \times (a_1 - a_0)) = \operatorname{sign}(e_1 \times (a_0 - a_1))$, and if $\operatorname{sign}(k_0) = \operatorname{sign}(e_0 \times e_1)$, assume that either

$$|k_0| < \frac{2(e_0 \times e_1)^2 |e_0 \times (a_1 - a_0)|}{3(e_1 \times (a_0 - a_1))^2}$$

$$|k_1| < \frac{2(e_0 \times e_1)^2 |e_1 \times (a_0 - a_1)|}{3(e_0 \times (a_1 - a_0))^2}$$

or

$$|k_0| > \frac{2(e_0 \times e_1)^2 |e_0 \times (a_1 - a_0)|}{3(e_1 \times (a_0 - a_1))^2}$$

$$|k_1| > \frac{2(e_0 \times e_1)^2 |e_1 \times (a_0 - a_1)|}{3(e_0 \times (a_1 - a_0))^2}.$$

Let it be mentioned that for peak-point curves, and also for point-point curves, point-tangent curves, and point curves, the assumption that $\mathrm{sign}(e_0 \times (a_1 - a_0)) = \mathrm{sign}(e_1 \times (a_0 - a_1))$ is for the purpose of avoiding that the Bezier curve has an inflection point, but the assumption is only a sufficient assumption not a necessary assumption.

The following formulas are provided for the five cases above:

1) For $\lambda$ and $\mu$

$$\lambda = \frac{3e_1 \times (a_o - a_1)}{e_0 \times e_1}$$
$$\mu = \frac{3e_0 \times (a_1 - a_0)}{e_0 \times e_1}$$

2) For $\lambda$ and $\mu$

$$\lambda = \frac{3e_1 \times (a_o - a_1)}{e_0 \times e_1}$$
$$\mu = \frac{3e_0 \times (a_1 - a_0)}{e_0 \times e_1} - k_0 \frac{9(e_1 \times (a_0 - a_1))^2}{2(e_0 \times e_1)^3}$$

3) For $\lambda$ and $\mu$

$$\lambda = \frac{3e_1 \times (a_o - a_1)}{e_0 \times e_1} - k_1 \frac{9(e_0 \times (a_1 - a_0))^2}{2(e_0 \times e_1)^3}$$
$$\mu = \frac{3e_0 \times (a_1 - a_0)}{e_0 \times e_1}$$

4) For $\lambda$ and $\mu$

$$\lambda = \sqrt{\frac{6e_0 \times (a_1 - a_0)}{k_0}}$$
$$\mu = \sqrt{\frac{6e_1 \times (a_0 - a_1)}{k_1}}$$

5) For $\lambda$ and $\mu$ the system of equations

$$f(\lambda, \mu) = k_0\lambda^2 + 2e_0 \times e_1\mu - 6e_0 \times (a_1 - a_0) = 0$$

$$g(\lambda, \mu) = k_1\mu^2 + 2e_0 \times e_1\lambda - 6e_1 \times (a_0 - a_1) = 0.$$

For case 5) there are the following solutions:

Let the Jacobian

$$J(\lambda,\mu) = \begin{pmatrix} \partial f/\partial\lambda & \partial f/\partial\mu \\ \partial g/\partial\lambda & \partial g/\partial\mu \end{pmatrix} = \begin{pmatrix} 2k_0\lambda & 2e_0 \times e_1 \\ 2e_0 \times e_1 & 2k_1\mu \end{pmatrix}$$

have a determinant unequal to 0 at the iterated solutions above. If sign($k_0$) = sign($e_0 \times e_1$), the following

$$\begin{pmatrix} \lambda_0 \\ \mu_0 \end{pmatrix} = \begin{pmatrix} \sqrt{(3e_0 \times (a_1 - a_0))/k_0} \\ \sqrt{(3e_1 \times (a_0 - a_1))/k_1} \end{pmatrix}$$

is taken as an initial solution. If sign($k_0$) = sign($e_0 \times e_1$), the following

$$\begin{pmatrix} \lambda_0 \\ \mu_0 \end{pmatrix} = \begin{pmatrix} \sqrt{(12e_0 \times (a_1 - a_0))/k_0} \\ \sqrt{(12e_1 \times (a_0 - a_1))/k_1} \end{pmatrix}$$

is taken as an initial solution. The system of equations for ($\lambda$, $\mu$) is solved through Newton's iteration method

$$\begin{pmatrix} \lambda_{i+1} \\ \mu_{i+1} \end{pmatrix} = \begin{pmatrix} \lambda_i \\ \mu_i \end{pmatrix} - J(\lambda_i, \mu_i)^{-1} \begin{pmatrix} f(\lambda_i, \mu_i) \\ g(\lambda_i, \mu_i) \end{pmatrix},$$

where

$$J(\lambda_i, \mu_i)^{-1} = \frac{1}{2(k_0 k_1 \lambda_i \mu_i - (e_0 \times e_1)^2)} \begin{pmatrix} k_1\mu_i & -e_0 \times e_1 \\ -e_0 \times e_1 & k_0\lambda_i \end{pmatrix}.$$

Figs. 36-38 — Straight Line Segments

A straight line segment may be drawn as a curve component of a curve. To draw it for example the shift key is held down when the end point is set for any of the types of curve component (or when the peak point is set if the curve is a point-point curve). Let the straight line segment be the first curve component. In a first step shown in Fig. 36, a start point $a_0$ and a start tangent direction $e_0$ are set. Start tangent direction $e_0$ is set in the direction of the desired line. In a second step shown in Fig. 37, an end point $a_1$ is set along start tangent direction $e_0$, e.g., by holding the shift key down. Since placing a cursor (not shown) exactly along the start tangent direction $e_0$ is difficult, end point $a_1$ is projected automatically from the cursor onto the half line corresponding to the start tangent direction $e_0$. A straight line segment $c_1$ is automatically drawn between start point $a_0$ and end point $a_1$. As shown in Fig. 38, an end tangent direction $e_1$ is automatically set in alignment with line $c_1$, but no peak point is set.

This straight line segment $c_1$ is already drawn when the mouse button is pressed for the end point $a_1$, and when the mouse is dragged, the end point $a_1$ is dragged to a new position along the start tangent direction $e_0$, and the straight line segment $c_1$ is changed, the final shape of which is drawn when the mouse is released.

If the straight line segment is the nth curve component after the first curve component and is connected with $G^2$ or $G^1$ continuity, only the end point $a_n$ has to be set along the start tangent direction $e_{n-1}$ (the end

tangent direction of the previous curve component). In the case of $G^2$ continuity, the end curvature of the previous curve component has to be 0. A straight line segment $c_n$ is automatically drawn between point $a_{n-1}$ (the end point of the previous curve component) and point $a_n$. An end tangent direction $e_n$ is automatically set in alignment with line $c_n$. If the straight line segment is connected with $G^0$ continuity, also the start tangent direction $e_{n-1}'$ has to be set before setting the end point $a_n$ along the start tangent direction $e_{n-1}'$ .

<u>Figs.39-41 — Circular Arcs</u>

A circular arc may be drawn as a curve component of a curve. To draw it for example the control key is held down when the end point is set for any of the types of curve component (or when the peak point is set if the curve component is a point-point curve). Let the circular arc be the first curve component. In a fist step shown in Fig. 39, a start point $a_0$ and a start tangent direction $e_0$ are set. In a second step shown in Fig. 40, an end point $a_1$ is set. A circular arc $c_1$ is automatically drawn between start point $a_0$ and end point $a_1$, tangent to $e_0$. As shown in Fig. 41, an end tangent direction $e_1$ is automatically set . If the curve component with which the circle was drawn is a peak-point curve, a point-point curve, a point-tangent or a point curve, a peak point $p_1$ is also automatically set. If the curve component with which the circle was drawn is a curvature curve, centers $m_0$ and $m_1$ of start and end curvature circles (which coincide) are also automatically set. The circular arc is symmetric with respect to the axis that is perpendicular to

the chord and goes through the center of the chord and the peak point $p_1$ and the curvature center $m_0$ lie on this axis.

This circular arc $c_1$ is already drawn when the mouse button is pressed for the end point $a_1$, and when the mouse is dragged, the end point $a_1$ is dragged to a new position, and the circular arc $c_1$ is changed, the final shape of which is drawn when the mouse is released.

If the circular arc is the nth curve component after the first curve component and is connected with $G^2$ continuity, only the end point $a_n$ has to be set. Because of $G^2$ continuity, the circular arc has to be part of the end curvature circle of the previous curve component. Since placing the cursor (not shown) exactly onto this curvature circle is difficult, end point $a_n$ is projected automatically from the cursor onto this curvature circle by being projected towards the center of this curvature circle. The circular arc is automatically drawn between point $a_{n-1}$ (the end point of the previous curve component) and point $a_n$ as part of the end curvature circle of the previous curve component and the center $m_n$ of the end curvature circle of the circular arc equals the center $m_{n-1}$ of the end curvature circle of the previous curve component. If the circular arc is connected with $G^1$ continuity, also only the end point $a_n$ has to be set. A circular arc is automatically drawn between point $a_{n-1}$ (the end point of the previous curve component) and point $a_n$, tangent to $e_{n-1}$ (the end tangent direction of the previous curve component). An end tangent direction $e_n$ and a peak point $p_n$ or centers $m_{n-1}$ and $m_n$ of the start and end curvature circles (which coincide) are automatically set. If

the circular arc is connected with $G^0$ continuity, also the start tangent direction $e'_{n-1}$ has to be set before setting the end point $a_n$, and the circular arc $c_n$ is tangent to $e'_{n-1}$.

It is possible to change a circular arc into a cubic Bezier curve by for example first selecting a button for "selections" and then selecting the end point while holding the "escape" key down. If the start or end point of the circular arc is connected with $G^2$ continuity, a curvature curve is computed with the start and end points, the start and end tangent directions of the circular arc, and the start and end curvatures equal to the curvature of the circular arc. Otherwise a peak-point curve is computed with the start and end points, the start and end tangent directions, and the peak point of the circular arc.

Also, it is possible to change any curve component (that is not a straight line segment) that is a peak-point curve, a point-point curve, a point-tangent curve, or a point curve into a circular arc by for example first selecting a button for "selections" and then selecting the end point while holding the "escape" key down. For a peak-point curve or a point-tangent curve this changes the end tangent direction and the peak point of the curve component, but the start and end points, and the start tangent direction remain unchanged. For a point-point curve or a point curve this only changes the peak point of the curve component, because the end tangent direction of a circular arc is also symmetric (i.e. the angle between the chord vector and the end tangent direction

equals the angle between the start tangent direction and the chord vector).

Continuity of Curve Components

In one embodiment, when drawing curves, all curve components except curvature curve components always connect with $G^1$ continuity by default. If the user wishes to connect a curve component with $G^0$ continuity, a button for $G^0$ continuity has to be selected first. In the same embodiment, when drawing curves, curvature curve components are always connected with $G^2$ continuity by default. If the user wishes to connect a curvature curve component by $G^1$ or $G^0$ continuity, a button for $G^1$ continuity or $G^0$ continuity has to be selected first. Curvature curves are the only curve components which can be connected with $G^2$ continuity.

The continuity how two curve components connect can be changed. To change $G^1$ continuity into $G^0$ continuity at an anchor point, the tangent direction at the anchor point can be split into two tangent directions by for example holding the "1" key down and selecting and changing the end tangent direction (or holding the "2" key down and selecting and changing the start tangent direction) at the anchor point. To change $G^0$ continuity into $G^1$ continuity, two split tangents are made one tangent by for example holding the "3" key down and selecting the tangent the user wants to keep.

To change $G^2$ continuity into $G^1$ continuity at an anchor point, the curvature at the anchor point can be split into two curvatures by for example holding the "1" key down and selecting and changing the center of the end curvature circle (or holding the "2" key down and selecting and changing the center of the start curvature circle) at the anchor point. To change $G^1$ continuity into $G^2$ continuity, two split curvatures can be made one curvature by for example holding the "3" key down and selecting the curvature one wants to keep.

The user can even change $G^0$ continuity into total discontinuity, because a curve can be split by for example first selecting the button for "moving", then holding the "1" key down and selecting and moving the curve ending at a selected anchor point (or holding the "2" key down and selecting and moving the curve starting at a selected anchor point). This splits the curve and keeps the moved curve selected, while it deselects the other curve. Also two separate curves can be made one curve by for example first selecting the button for "moving", then holding the "3" key down and selecting and moving a curve from a selected start point close to an end point of another curve (or selecting and moving a curve from a selected end point close to a start point of another curve), and the two curves connect with $G^0$ continuity.

While drawing a curve (or after selecting it after for example selecting a button for "selections"), the last curve component can be deleted by for example pressing the "delete" or "back-space" key. If the "delete" key is pressed again, the second to the last curve component of the original

curve is deleted, and then the third to the last curve component, and so on. If, after deleting one or more curve components, the user wants to continue drawing the curve, a button for the "curve type" has to be selected first.

Now, because it is possible to make two curves out of one curve, and one curve out of two curves, and because it is possible to change the continuity how two curve components connect, the user can delete and redraw not only the last curve component, but any curve component. To do that for a curve component which is not the last curve component, the user first splits the curve at the end point of the chosen curve component, then deletes that curve component, then redraws that curve component, then again makes one curve out of the two curves, which connects them with $G^0$ continuity, and then changes the continuity to $G^1$ continuity is desired, and last changes the continuity to $G^2$ continuity if desired..

<u>Modifying Features and Types of Curve Components</u>

In one embodiment for selecting a feature a mouse, or other input device is used. While drawing a curve, if some features are already drawn for the construction of the next curve component, which has not been drawn yet, any such feature can be deleted by pressing the "delete" key (starting with the last feature drawn, proceeding with the second the last feature, and so on, and the position of any such feature can be changed by for example first selecting a button for "selections" and then selecting the feature.

While drawing a curve (or after selecting it after selecting a button for "selections"), the position of any feature of a curve component that has been drawn can also be changed by for example first selecting a button for "selections" and then selecting the feature. When the feature changes, the curve changes, because the system automatically adapts the shape of the curve to the changed feature.

Each curve component is identified as a curve component of a particular type, and when the position of a feature of the curve component is changed, the curve component changes as a curve component of that type. However, although there are 7 different constructions of curve components, there are only 5 types of curve components remaining by which a curve component is identified for the purpose of making changes: peak-point curves, point-tangent curves, curvature curves, straight line segments, and circular arcs. A curve component that was constructed as a straight line segment or as a circular arc always is identified as a curve component of the same type. A curve component that was constructed as a peak-point curve or as a point-point curve always is identified as a peak-point curve unless it is specifically made a point-tangent curve as described below. A curve component that was constructed as a point-tangent curve or as a point curve is identified as a point-tangent curve as long as the peak point has not been changed, but as soon as the peak point has been changed for the first time, it is identified as a peak-point curve. A curve component that was constructed as a curvature curve only is identified as a curvature curve

as long as it is connected with $G^2$ continuity at the start or end point, otherwise it is identified as a peak-point curve.

However, it is possible to change a curve component that is identified as a peak-point curve into a point-tangent curve by for example holding the "escape" key down and selecting the peak point. This computes automatically the point-tangent curve with the same start and end points and the same start and end tangents, and the peak point is changed to the peak point of the point-tangent curve (which is computed as a quadratic Bezier curve). It is also possible to change a point-tangent curve into a peak-point curve by slightly changing the position of the peak point.

When the position of a feature is changed, the type of continuity by which the curve component is connected at the start and end points remains the same. The type of features that can be changed for a curve component depend on the type of the curve component. For a straight line segment only the start or end points, or start or end tangent directions can be changed. For a circle, the start or end point, or start or end tangent directions, and curvature can be changed, and if it is not connected with $G^2$ continuity at the start and end points, the peak point can be changed. For a curve component of the type of a curvature curve only the start or end point, the start or end tangent direction, or the start and end curvature can be changed. (The peak point cannot be changed, because this would interfere with the $G^2$ continuity.) For a curve component of the type of a peak-point curve or point-tangent curve, all features can be changed: the start and end points, the start

and end tangent directions, the start and end curvatures, and the peak point. However, the start and end curvatures are not shown unless a special key is held down such as the "c" key. If the "c" key is held down, the curve component changes as a curvature curve (which means that the start and end curvatures can be changed, and also that changes for the start or end points or start or end tangent directions produce a different change in the curve component than if the "c" key is not held down).

If the "c" key is not held down, the curve component changes as a peak-point curve or a point-tangent curve. If the curve component changes as a point-tangent curve, the peak point gets changed automatically, when the start point, end point, start tangent direction, or end tangent direction is changed (because the quadratic Bezier curve is changed, which gives a new peak point). This gives more flexibility for those changes, whereas for peak-point curves, since the peak point does not get changed with those changes, there is less flexibility.

If a curve component is a straight line segment, it responds differently when a feature gets changed than if the curve component is a Bezier curve, because when one feature is changed, other features are changed also automatically. When the position of the start or end point is changed, both tangent directions are changed automatically, but the other point remains unchanged. When the position of the start or end tangent direction is changed, the other point and other tangent direction

are changed automatically, but the corresponding point and the length of the straight line segment remain unchanged.

If a curve component is a circular arc, it responds differently when a feature gets changed than if the curve component is a Bezier curve, because when one feature is changed, other features are changed also automatically. When the position of the start or end point is changed, the corresponding tangent direction and the peak point are changed automatically, but the other point and tangent direction remain unchanged. When the position of the start or end tangent direction is changed, the other point and tangent direction and the peak point are changed automatically, but the corresponding point remains unchanged. When the position of the peak point is changed, the start and end tangent directions are changed, but the start and end points remain unchanged.

If the curve component is not a point-tangent curve, not a straight line segment, and not a circular arc, when any feature of the curve component is changed, only that feature is changed and the other features of the curve component remain unchanged. Also when the peak point of a point-tangent curve is changed, the other features remain unchanged.

If the feature is a peak point, no other curve component gets changed. However if the feature is an anchor point, both curve components which connect at that anchor point are changed. If the feature is a tangent

direction at an anchor point, and if the curve components are connected with $G^1$ or $G^2$ continuity at that anchor point, both curve components are changed, but if they are connected with $G^0$ continuity, only the curve component of the selected tangent direction is changed. If the feature is a curvature at an anchor point, and if the curve components are connected with $G^2$ continuity at that anchor point, both curve components are changed, but if they are connected with $G^1$ or $G^0$ continuity only the curve component of the selected curvature is changed.

Adding and Subtracting Curve Components

For a general curve, that is composed of curve components of any curve-type, curve components can be added or subtracted with any suitable method. A curve component can be added by for example first selecting a button for "additions" and then selecting a point on the curve between a start and an end point of a curve component. Then this curve component becomes two curve components which connect at this point. If the original curve component is a Bezier curve, the two curve components are simply portions of this Bezier curve, which are reparametereized so that the parameter interval is [0, 1]. If the original curve component is a circular arc, the two curve components are portions of this circular arc and have the same radius. In both cases the selected point and tangent direction is shown. If the original curve component is of the type of a peak-point curve or point-tangent curve the peak points of the two curve components are shown. To compute the peak point of a cubic Bezier curve x(t) with start and end points $a_0$

and $a_1$, the following quadratic equation for the parameter t of the peak point is solved:

$$x'(t) \times (a_1 - a_0) = 0,$$

where "x" denotes the vector product between two 2-dimensional vectors. If the original curve component is of the type of a curvature curve, the center of the curvature circle at the selected point is shown together with the curvature line, and the curvature lines at the start and end points of the curve component are adjusted.

For a general curve, that is composed of curve components of any curve-type, curve components can also be subtracted by for example first selecting a button for "subtractions" and then selecting an anchor point on the curve. Then the two curve components which connect at that anchor point become one curve component. If neither of the two curve components is of the type of a curvature curve, the one curve component is computed as a peak-point curve which has the same start point $a_0$ and the same start tangent direction as the start point and the start tangent direction of the first curve component and the same end point $a_1$ and the same end tangent direction as the end point and end tangent direction of the second curve component and the peak point equals the point on one of the two curve components which is tangent to a1 – a0. In general, to compute the point on a cubic Bezier curve x(t) that is tangent to a given vector v, the following quadratic equation for the parameter t of the point is solved:

$$x'(t) \times v = 0,$$

where "x" denotes the vector product between two 2-dimensional vectors. If at least one of the two curve components is of the type of a curvature curve, the one curve component is computed as a curvature curve which has the same start point, the same start tangent direction, and the same start curvature as the start point, the start tangent direction, and the curvature of the first curve component and the same end point, the same end tangent direction, and the same end curvature as the end point, end tangent direction, and end curvature of the second curve component.

The additions always work, but, in one embodiment, the subtractions do not work if the assumptions described above for computing the peak-point curve or the curvature curve are not fulfilled.

<u>Computing Points and Tangent Directions of a Quadratic Bezier Curve</u>

Let $b_0$, $b_1$, and $b_2$ be the control points of a quadratic Bezier curve. For drawing a quadratic Bezier curve or for selecting a point on a quadratic Bezier curve the deCasteljau algorithm with respect to parameter 1/2 is used. This algorithm is faster than making computations using the equation of the quadratic Bezier curve. It computes a number N of points and tangent directions, the number N being a power of 2 (such as N = 1024). For drawing the quadratic Bezier curve, the polygonal line connecting these points is drawn. The points are not computed in the order they are drawn, but each point has an index and the points are drawn in the order of the indices. The points p[n], n = 0, 1, 2, ..., N, start at the start point $p[0] = b_0$ and end at the end point $p[N] = b_2$. The tangent directions t[n], n = 0, 1, 2, ..., N, start at the start tangent

direction t[0] = vector from $b_0$ to $b_1$, and end at the end tangent direction t[N] = vector from $b_1$ to $b_2$. The points p[n] and tangent directions t[n], n = 1, 2, 3, ..., N – 1 are computed by a recursive function depending on an index n, a step s, and control points $c_0$, $c_1$, and $c_2$ (for a quadratic Bezier curve which is a portion of the original quadratic Bezier curve). This recursive function also depends on the arrays p[ ] and t[ ], where it writes the results. The first time this function is called for the index N/2, the step N/2, and the control points $c_0 = b_0$, $c_1 = b_1$, and $c_2 = b_2$, and it computes the point p[N/2] and the tangent direction t[N/2]. When the function is called for the index n it computes the point

$$p[n] = 0.5*(0.5*(c_0 + c_1) + 0.5*(c_1 + c_2))$$

and the tangent direction

$$t[n] = \text{vector from } 0.5*(c_0 + c_1) \text{ to } 0.5*(c_1 + c_2).$$

After p[n] and t[n] are computed, if s is still larger than 1, the recursive function gets called twice for the index n – s/2, the step s/2, and the control points

$$c_0,\ 0.5*(c_0 + c_1), \text{ and } p[n]$$

(for the quadratic Bezier curve between $c_0$ and p[n]),

and for the index n + s/2, the step s/2, and the control points

$$p[n],\ 0.5*(c_1 + c_2), \text{ and } c_2$$

(for the quadratic Bezier curve between p[n] and $c_2$).

The point p[n] and tangent direction t[n] are the point and tangent direction at the parameter t = n/N, and they are computed faster than using the equation of the original Bezier curve.

Computing Points and Tangent Directions of a Cubic Bezier Curve

Let $b_0$, $b_1$, $b_2$, and $b_3$ be the control points of a cubic Bezier curve. For drawing a cubic Bezier curve or for selecting a point on a cubic Bezier curve the deCasteljau algorithm with respect to parameter 1/2 is used. This algorithm is faster than making computations using the equation of the cubic Bezier curve. It computes a number N of points and tangent directions, the number N being a power of 2 (such as N = 1024). For drawing the cubic Bezier curve, the polygonal line connecting these points is drawn. The points are not computed in the order they are drawn, but each point has an index and the points are drawn in the order of the indices. The points p[n], n = 0, 1, 2, ..., N, start at the start point p[0] = $b_0$ and end at the end point p[N] = $b_3$. The tangent directions t[n], n = 0, 1, 2, ..., N, start at the start tangent direction t[0] = vector from $b_0$ to $b_1$ and end at the end tangent direction t[N] = vector from $b_2$ to $b_3$. The points p[n] and tangent directions t[n], n = 1, 2, 3, ..., N – 1 are computed by a recursive function depending on an index n, a step s, and control points $c_0$, $c_1$, $c_2$, and $c_3$ (for a cubic Bezier curve which is a portion of the original cubic Bezier curve). This recursive function also depends on the arrays p[ ] and t[ ], where it writes the results. The first time this function is called for the index N/2, the step N/2, and the control points $c_0 = b_0$, $c_1 = b_1$, $c_2 = b_2$, and $c_3 = b_3$, and it computes the point p[N/2] and the tangent direction t[N/2]. When the function is called for the index n it computes the point

$$p[n] = 0.5*(0.5*(0.5*( c_0 + c_1) + 0.5*( c_1+ c_2)) + 0.5*(0.5*( c_1 + c_2) + 0.5*( c_2+ c_3)))$$

and the tangent direction

$$t[n] = \text{vector from } 0.5^*(0.5^*( c_0 + c_1) + 0.5^*( c_1 + c_2))$$
$$\text{to } 0.5^*(0.5^*( c_1 + c_2) + 0.5^*( c_2 + c_3)).$$

After p[n] and t[n] are computed, if s is still larger than 1, the recursive function calls itself twice for the index n – s/2, the step s/2, and the control points

$$c_0,\ 0.5^*( c_0 + c_1),\ 0.5^*(0.5^*( c_0 + c_1) + 0.5^*( c_1 + c_2)), \text{ and } p[n]$$

(for the cubic Bezier curve between $c_0$ and p[n]),

and for the index n + s/2, the step s/2, and the control points

$$p[n],\ 0.5^*(0.5^*( c_1 + c_2) + 0.5^*( c_2 + c_3)),\ 0.5^*( c_2 + c_3), \text{ and } c_3$$

(for the cubic Bezier curve between p[n] and $c_3$).

The point p[n] and tangent direction t[n] are the point and tangent direction at the parameter t = n/N, and they are computed faster than using the equation of the original Bezier curve.

Conclusion

Although the above description is specific, it should not be considered as a limitation on the scope of the invention, but only as an example of the preferred embodiment. Many substitutes and variations are possible within the teachings of the invention. The mathematical formulas are only examples how the curves may be determined; any other suitable formulas may be used. The order in which the points, tangent directions, and peak points are set may be different.

In one embodiment, for all curves, peak-point curves, point-point curves, point-tangent curves, and point curves, each curve component is curved

in only one direction, i.e., it has no inflection point. In this embodiment an inflection point must be constructed explicitly by connecting two curve components that are curved in opposite directions, e.g., in Fig. 11, Fig. 18, Fig. 23, and Fig. 28 the point $a_1$ is an inflection point.

In one embodiment, any feature that is drawn — be it a point, a tangent direction, or a peak point — is shown immediately while the mouse button is pressed. However, if the feature cannot be drawn because of the assumptions in the embodiment (such as the assumptions described above together with the formulas), it is not drawn, and the cursor has to be positioned somewhere else where the feature can be drawn. There might be less freedom how a feature can be positioned than in a typical computer drawing program, however there is more control how the features determine the curve.

While the method and apparatus of the present invention has been described in terms of its presently preferred and alternate embodiments, those skilled in the art will recognize that the present invention may be practiced with modification and alteration within the spirit and scope of the appended claims. The specifications and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense. Further, even though only certain embodiments have been described in detail, those having ordinary skill in the art will certainly understand that many modifications of the embodiments are possible without departing from the teachings thereof. All such modifications are intended to be encompassed within the following claims.

CLAIMS

I claim:

1. A computer curve construction system for constructing a curve that includes a plurality of curve components that are connected with a certain type of continuity, the system comprising:

a user input device; and

a processor coupled to the input device, and wherein the processor responds to signals received from the input device to determine a first curve component, the processor being operative to define each curve component after the first curve component as connected to the previous curve component with continuity that may be either geometric order 0 (G0) or geometric order 1 (G1).

2. The computer curve construction system of claim 1, and where in the processor responds to feature modification signals provided by the input device by modifying curve features of a selected curve component.

3. The computer curve construction system of claim 1, and wherein the processor responds to continuity modification signals provided by the input device by modifying the continuity between selected curve components.

4. The computer curve construction system of claim 1, and wherein a selected curve component is selected from the group consisting of a peak-point curve, a point-point curve, a point-tangent curve and a point curve.

5. The computer curve construction system of claim 1, and wherein a selected curve component comprises a circular curve.

6. A computer curve construction method for constructing a curve that includes a plurality of curve components that are connected with a certain type of continuity, the method comprising:

determining a first curve component; and

defining each curve component after the first curve component as connected to the previous curve component with continuity that may be either geometric order 0 (G0) or geometric order 1 (G1).

7. The computer curve construction method of claim 6, and further comprising:

modifying curve characteristics of a selected curve component.

8. The computer curve construction method of claim 6, and further comprising:

modifying the continuity between selected curve components.

9. The computer curve construction method of claim 6, and wherein a selected curve component is selected from the group consisting of a peak-point curve, a point-point curve, a point-tangent curve and a point curve.

10. The computer curve construction method of claim 6, and wherein a selected curve component comprises a circular arc.

11: A computer curve construction system for constructing a curve that includes a plurality of connected curve components, the system comprising:

an input device; and

a processor coupled to the user input device, and wherein the processor responds to signals received from the input device to determine each curve component as having a start point, a start tangent direction at the start point, an end point spaced from the start point and an end tangent direction at the end point, the processor being operative to define each curve component after a first curve component as having a start point equal to the end point of the previous curve component and a start tangent direction equal to the end tangent direction of the previous curve component.

12. The computer curve construction system of claim 11, and further comprising:

a display device connected to the processor for displaying the curve components.

13. A computer curve construction method for constructing a curve that includes a plurality of connected curve components, the method comprising:

defining a first curve component having a start point, a start tangent direction at the start point, an end point spaced from the start point and an end tangent direction at the end point; and

defining each curve component after the first curve component as having a start point equal to the end point of the previous curve component and a start tangent direction equal to the end tangent direction of the previous curve component.

14. The method of claim 13, and further comprising:

displaying the continuous curve components utilizing a display device.

15. A computer curve construction system for constructing peak-point curves, the system comprising:

an input device; and

a processor coupled to the input device, and wherein the processor responds to signals received from the input device to determine a start point, a start tangent direction at the start point, an end point spaced from the start point, an end tangent direction at the end point and a peak point spaced from the start point and the end point, the processor being operative to generate a curve between the start point and the end point and passing through the peak point, the curve being defined by a mathematical formula using the start point, the start tangent direction, the end point, the end tangent direction and the peak point as variables, and wherein the peak point defines the largest distance between the curve and a chord connecting the start point and the end point.

16. The computer curve construction system of claim 15, and wherein the processor responds to additional signals received from the input device to determine an additional end point spaced from the end point, an additional end tangent direction at the additional end point, an additional peak point spaced from the end point and the additional end point, and is further operative to generate an additional curve between the end point and the additional end point and passing through the additional peak point, the additional curve being defined by the

mathematical formula using the end point, the end tangent direction, the additional end point, the additional end tangent direction and the additional peak point as variables, and wherein the additional peak point defines the largest distance between the additional curve and a chord connecting the end point and the additional end point.

17. The computer curve construction system of claim 15, and wherein the curve comprises a cubic Bezier curve.

18. The computer curve construction system of claim 15, and further comprising:

a display device connected to the processor for displaying the curve.

19. A computer curve construction method for constructing a peak-point curve, the method comprising:

determining a start point;

determining a start tangent direction at the start point;

determining an end point spaced from the start point;

determining an end tangent direction at the end point;

determining a peak point spaced from the start point and the end point; and

generating a curve between the start point and the end point and passing through the peak point, the curve being defined by a mathematical formula using the start point, the start tangent direction, the end point, the end tangent direction and the peak point as variables,

and wherein the peak point defines the largest distance between the curve and a chord connecting the start point and the end point.

20. The computer curve construction method of claim 19, and further comprising:

determining an additional end point spaced from the end point;

determining an additional end tangent direction at the additional end point;

determining an additional peak point spaced from the end point and the additional end point; and

generating an additional curve between the end point and the additional end point and through the additional peak point, the additional curve being defined by the mathematical formula using the end point, the end tangent direction, the additional end point, the additional end tangent direction and the additional peak point as variables, the additional peak point defining the largest distance between the additional curve and an additional chord connecting the end point and the additional end point.

21. The computer curve construction method of claim 19, and further comprising:

displaying the curve utilizing a display device.

22. The computer curve construction method of claim 19, and wherein the curve comprises a cubic Bezier curve.

23. A computer curve construction system for constructing point-point curves, the system comprising:

an input device; and

a processor coupled to the input device, and wherein the processor responds to signals provided by the input device to determine a start point, a start tangent direction at the start point, an end point

spaced from the start point and a peak point spaced from the start point and the end point, the processor being operative to generate a curve between the start point and the end point and passing through the peak point, the curve being defined by a mathematical formula using the start point, the start tangent direction, the end point and the peak point as variables, the peak point defining the largest distance between the curve and a chord connecting the start point and the end point.

24. The computer curve construction system of claim 23, and wherein the processor responds to additional signals provided by the input device to determine a new peak point and to generate a new curve passing through the start point, the end point and the new peak point, the new curve being defined by the mathematical formula using the start point, the start tangent direction, the end point and the new peak point as variables.

25. The computer curve construction system of claim 23, and wherein the processor responds to additional signals provided by the input device to change the shape of the curve in response to a change in the start tangent direction, and wherein the distance between the peak point and the chord connecting the start point and the end point remains constant.

26. The computer curve construction system of claim 23, and wherein the curve defines an end tangent direction at the end point, and wherein the start tangent direction and the end tangent direction are related.

27. The computer curve construction system of claim 26, and wherein the angle between the start tangent direction and the chord equals the angle between the chord and the end tangent direction.

28. The computer curve construction system of claim 26, and wherein the angle between the start tangent direction and the chord is a preselected integer or fractional multiple of the angle between the chord and the end tangent direction.

29. The computer curve construction system of claim 23, and wherein the curve comprises a cubic Bezier curve.

30. The computer curve construction system of claim 23, and wherein the curve defines an end tangent direction at the end point and wherein the processor responds to additional signals provided by the input device to determine an additional end point spaced from the end point and an additional peak point spaced from the end point and the additional end point, and is further operative to generate an additional curve between the end point and the additional end point and passing through the additional peak point, the additional curve being defined by the mathematical formula using the end point, the end tangent direction, the additional end point and the additional peak point as variables, the additional peak point defining the largest difference between the additional curve and a chord between the end point and the additional end point.

31. A computer curve construction system for constructing point-point curves, the system comprising:

an input device; and

a processor coupled to the input device, and wherein the processor responds to signals provided by the input device to determine a start point, an end point spaced from the start point and a peak point spaced from the start point and the end point, the processor being operative to generate a curve between the start point and the end point and passing through the peak point, the curve being a quadratic Bezier curve defined by a mathematical formula using the start point, the end point and the peak point as variables, the peak point defining the largest distance between the curve and a chord connecting the start point and the end point.

32. The computer curve construction system of claim 31, and wherein the processor responds to additional signals provided by the input device to determine an additional end point spaced from the end point and an additional peak point spaced from the end point and the additional end point, and is further operative to generate an additional curve between the end point and the additional end point passing through the additional peak point, the additional curve being defined by the mathematical formula using the end point, the additional end point and the additional peak point as variables, the additional peak point defining the largest distance between the additional curve and a chord between the end point and the additional end point.

33. A computer curve construction method for constructing point-point curves, the method comprising:

determining a start point;

determining a start tangent direction at the start point;

determining an end point spaced from the start point;

determining a peak point spaced from the start point and from the end point; and

generating a curve between the start point and the end point, the curve being defined by a mathematical formula using the start point, the start tangent direction, the end point and the peak point as variables, the peak point defining the largest distance between the curve and a chord connecting the start point and the end point.

34. The computer curve construction method of claim 33, and further comprising:

changing the start tangent direction;

changing the shape of the curve in response to the changed start tangent direction; and

maintaining the distance between the peak point and the chord connecting the start point and the end point constant when changing the shape of curve.

35. The computer curve construction method of claim 33, and further comprising:

changing the peak point to define a new peak point; and

generating a new curve passing through the start point, the end point and the new peak point, the new curve being defined by the mathematical formula using the start point, the start tangent direction, the end point and the new peak point as variables.

36. The computer curve construction method of claim 33, and wherein the curve defines an end tangent direction at the end point, and wherein the start tangent direction and the end tangent direction are related.

37. The computer curve construction method of claim 36, and wherein the angle between the start tangent direction and the chord equals the angle between the chord and the end tangent direction.

38. The computer curve construction method of claim 36, and wherein the angle between the start tangent direction and the chord is a preselected integer or fractional multiple of the angle between the chord and the end tangent direction.

39. The computer curve construction method of claim 33, and wherein the curve comprises a cubic Bezier curve.

40. The computer curve construction method of claim 33, and wherein the curve defines an end tangent direction at the end point, and further comprising:

determining an additional end point spaced from the end point;

determining an additional peak point spaced from the end point and from the additional end point; and

generating an additional curve between the end point and the additional end point and passing through the additional peak point, the additional curve being defined by the mathematical formula using the end point, the end tangent direction, the additional end point and the additional peak point as variables, the additional peak point defining the largest distance between the additional curve and a chord between the end point and the additional end point.

41. A computer curve construction method for constructing point-point curves, the method comprising:

determining a start point;

determining an end point spaced from the start point;

determining a peak point spaced from the start point and the end point; and

generating a curve between the start point and the end point and passing through the peak point, the curve being a quadratic Bezier curve defined by a mathematical formula using the start point, the end point and the peak point as variables, the peak point defining the largest distance between the curve and a chord connecting the start point and the end point.

42. A computer curve construction system, comprising:

an input device; and

a processor coupled to the input device, wherein the processor responds to signals received from the input device to determine a start point, a start tangent direction at the start point, an end point spaced from the start point and a first angle between the start tangent direction and a chord between the start point and the end point, the processor being operative to automatically set an end tangent direction at the end point at a second angle related to the first angle, to identify a peak point spaced at equal distances from the start point and the end point according to a first mathematical formula, and to generate a curve between the start point and the end point and passing through the peak point, the curve being defined by a second mathematical formula using the start point, the start tangent direction, the end point and the end tangent direction as variables, the peak point defining the largest distance between the curve and the chord.

43. The computer curve construction system of claim 42, and wherein the processor responds to additional signals provided by the input

device to define an additional end point spaced from the end point, and to determine a third angle between the end tangent direction and an additional chord between the end point and the additional end point, the processor being further operative to set an additional end tangent direction at the additional end point at a fourth angle to the additional chord which is related to the third angle, and to set an additional peak point spaced at equal distances from the end point and the additional end point according to the first mathematical formula, and to draw an additional curve between the end point and the additional end point and passing through the additional peak point, the additional curve being defined by the second mathematical formula using the end point, the end tangent direction, the additional end point and the additional end tangent direction as variables, the additional peak point defining the largest distance between the additional curve and a chord between the end point and the additional end point.

44. The computer curve construction system of claim 42, and wherein the curve comprises a quadratic Bezier curve.

45. A computer curve construction method for constructing a curve, the method comprising:

determining a start point;

determining a start tangent direction at the start point;

determining an end point spaced from the start point,

determining a first angle between the start tangent direction and a chord between the start point and the end point;

setting an end tangent direction at the end point at a second angle to the chord, the end tangent direction being related to the first angle;

determining a peak point spaced at equal distances from the start point and the end point according to a first mathematical formula; and

generating a curve between the start point and the end point and passing through the peak point, the curve being defined by a second mathematical formula using the start point, the start tangent direction, the end point and the end tangent direction as variables, the peak point defining the largest distance between the curve and the chord.

46. The computer curve construction method of claim 45, and further comprising:

determining an additional end point spaced from the end point;

determining a third angle between the end tangent direction and an additional chord between the end point and the additional end point;

determining an additional end tangent direction at the additional end point at a fourth angle to the additional chord, the fourth angle being related to the third angle;

determining an additional peak point spaced at equal distances from the end point and the additional end point according to the first mathematical formula; and

generating an additional curve between the end point and the additional end point and passing through the additional peak point according to the second mathematical formula.

47. The computer curve construction method of claim 45, and wherein the curve comprises a quadratic Bezier curve.

48. A computer curve construction system for constructing point-tangent curves, the system comprising:

an input device; and

a processor coupled to the input device, and wherein the processor responds to signals received from the input device to determine a start point, a start tangent direction at the start point, an end point spaced from the start point and an end tangent direction at the end point, the processor being operative to generate a curve between the start point and the end point, the curve being defined by a mathematical formula using the start point, the start tangent direction, the end point and the end tangent direction as variables.

49. The computer curve construction system of claim 48, and wherein the processor responds to additional signals received from the input device to determine an additional end point spaced from the end point and an additional end tangent direction at the additional end point, and is further operative to generate an additional curve between the end point and the additional end point, the additional curve being defined by the mathematical formula using the end point, the end tangent direction, the additional end point and the additional end tangent direction as variables.

50. The computer curve construction system of claim 48, and wherein the curve includes a peak point, the peak point defining the largest distance between the curve and a chord connecting the start point and the end point.

51. The computer curve construction system of claim 48, and wherein the curve comprises a quadratic Bezier curve.

52. A computer curve construction method for constructing a point-tangent curve, the method comprising:

determining a start point;

determining a start tangent direction at the start point;

determining an end point,

determining an end tangent direction at the end point; and

generating a curve between the start point and the end point, the curve being defined by mathematical formula using the start point, the start tangent direction, the end point and the end tangent direction as variables.

53. The computer curve construction method of claim 52, and further comprising:

determining an additional end point spaced from the end point;

determining an additional end tangent direction at the additional end point; and

generating an additional curve between the end point and the additional end point, the additional curve being defined by the mathematical formula using the end point, the end tangent direction, the additional end point and the additional end tangent direction as variables.

54. The computer curve construction method of claim 52, and wherein the curve includes a peak point, the peak point defining the largest distance between the curve and a chord connecting the end point and the end point.

55. The computer curve construction method of claim 52, and wherein the curve comprises a quadratic Bezier curve.

56. A computer curve construction system for constructing point curves, the system comprising:

an input device; and

a processor coupled to the input device, and wherein the processor responds to signals received from the input device to determine a start point, a start tangent direction at the start point and an end point, the processor being operative to generate a curve between the start point and the end point, the curve being defined by a mathematical formula using the start point, the start tangent direction and the end point as variables.

57. The computer curve construction system of claim 56, and wherein the curve defines an end tangent direction at the end point, and wherein the processor responds to additional signals received from the input device to determine an additional end point spaced from the end point, and is further operative to generate an additional curve between the end point and the additional end point, the additional curve being defined by the mathematical formula using the end point, the end tangent direction and the additional end point as variables.

58. The computer curve construction system of claim 56, and wherein the curve defines an end tangent direction at the end point, and wherein the start tangent direction and the end tangent direction are related.

59. The computer curve construction system of claim 58, and wherein the angle between the start tangent direction and a chord connecting the start point and the end point equals the angle between the chord and the end tangent direction.

60. The computer curve construction system of claim 58, and wherein the angle between the start tangent direction and a chord between the start point and the end point is a preselected integer or fractional multiple of the angle between the chord and the end tangent direction.

61. The computer curve construction system of claim 58, and wherein the curve comprises a quadratic Bezier curve.

62. A computer curve construction method for constructing a point curve, the method comprising:

determining a start point;

determining a start tangent direction at the start point;

determining an end point; and

generating a curve between the start point and the end point, the curve being defined by a mathematical formula using the start point, the start tangent direction and the end point as variables.

63. The computer curve construction method of claim 62 and wherein the curve defines an end tangent direction at the end point, and further comprising:

determining an additional end point spaced from the end point; and

generating an additional curve component between the end point and the additional end point, the additional curve component being defined by the mathematical formula using the end point, the end tangent direction and the additional end point as variables.

64. The computer curve construction method of claim 62, and wherein the curve defines an end tangent direction at the end point and

wherein the start tangent direction and the end tangent direction are related.

65. The computer curve construction method of claim 64, and wherein the angle between the start tangent direction and chord between the start point and the end point equals the angle between the chord and the end tangent direction.

66. The computer curve construction method of claim 64, and wherein the angle between the start tangent direction and a chord between the start point and the end point is a pre-selected integer or fractional multiple of the angle between the chord and the end tangent direction.

67. The computer curve construction system of claim 62, and wherein the curve comprises a quadratic Bezier curve.

68. The computer curve construction system of claim 11, and wherein the plurality of continuous curve components includes at least one circular arc.

69. The computer curve construction system of claim 68, and wherein the processor responds to circular arc generation signals received from the input device to define a start point for the circular arc, start tangent direction for the circular arc at the start point and an end point for the circular arc, and wherein the processor is further operative to define the circular arc using a circular arc mathematical formula having the circular arc start point, the circular arc tangent direction and the circular arc end point as variables.

70. The computer curve construction system of claim 68, and wherein the at least one circular arc is designated by a circular arc designation signal generated by the input device.

71. The computer curve construction system of claim 68, and wherein the processor responds to additional signals received from the input device to determine an additional circular arc end point spaced from the circular arc end point, and is further operative to generate an additional circular arc between the circular arc end point and the additional circular arc end point using the circular arc end point and the additional circular arc end point as variables.

72. The computer curve construction system of claim 11, and wherein the plurality of continuous curve components includes at least one straight line segment.

73. The computer curve construction system of claim 72, and wherein the processor responds to straight line segment generation signals received from the input device to define a start point for the straight line segment and an end point for the straight line segment, and wherein the processor is further operative to define the straight line segment using a straight line segment mathematical formula having the straight line segment start point and the straight line segment end point as variables.

74. The computer curve construction system of claim 72, and wherein the straight line segment is designated by a straight line segment designation signal generated by the input device.

75. The computer curve construction system of claim 72, and wherein the processor responds to additional signals received from the input device to determine an additional straight line segment end point spaced from the straight line segment end point, and is further operative to generate an additional straight line segment between the straight line segment end point and the additional straight line segment end point using the straight line segment end point and the additional straight line segment end point as variables in the straight line segment mathematical formula.

76. A computer curve construction system for constructing a curve that includes a plurality of curve components that are connected with a certain type of continuity, the system comprising:

an input device; and

a processor coupled to the input device, and wherein the processor responds to signals received from the input device to determine a first curve component, the processor being operative to define each curve component after the first curve component as connected to the previous curve component with continuity that may be either geometric order (G0) or geometric order 1 (G1) or geometric order 2.

77. The computer curve construction system of claim 76, and where in the processor responds to feature modification signals provided by the input device by modifying curve features of at a selected curve component.

78. The computer curve construction system of claim 76, and wherein the processor responds to continuity modification signals provided by

the input device by modifying the continuity between selected curve components.

79. A computer curve construction method for constructing a curve that includes a plurality of curve components that are connected with a certain type of continuity, the method comprising:

determining a first curve component; and

defining each curve component after the first curve component as connected to the previous curve component with continuity that may be either geometric order 0 (G0) or geometric order 1 (G1) or geometric order 2 (G2).

80. The computer curve construction method of claim 79, and further comprising:

modifying curve characteristics of a selected curve component.

81. The computer curve construction method of claim 79, and further comprising:

modifying the curve continuity of a selected curve component.

82. A computer curve construction system for constructing a curve comprising a plurality of connected curve components, the system comprising:

a user input device;

a processor coupled to the user input device, and wherein the processor responds to signals received from the input device to determine each curve component as having a start point, a start tangent direction at the start point, a start curvature at the start point, an end point, an end tangent direction at the end point and an end curvature at

the end point, the processor being operative to define each curve component after a first curve component as having a start point equal to the end point of the previous curve component, a start tangent direction at the start point equal to the end tangent direction at the end point of the previous curve component and a start curvature at the start point equal to the end curvature at the end point of the previous curve component.

83. A system of claim 82, and further comprising:

a display device connected to the processor for displaying the curve components.

84. A computer curve construction method for constructing a curve that includes a plurality of connected curve components, the method comprising:

defining a first curve component having a start point, a start tangent direction at the start point, a start curvature at the start point, an end point spaced from the start point, an end tangent direction at the end point and an end curvature at the end point; and

defining each curve component after the first curve component as having a start point equal to the end point of the previous curve component, a start tangent direction equal to the end tangent direction of the previous curve component and a start curvature equal to the end point curvature of the previous curve component.

85. The method of claim 84, and further comprising:

displaying the continuous curve component utilizing a display device.

86. A computer curve construction system for constructing curvature curves, the system comprising:

an input device; and

a processor coupled to the input device, wherein the processor responds to signals received from the input device to determine a start point, a start tangent direction at the start point, a start curvature at the start point, an end point spaced from the start point, an end tangent direction at the end point and a end curvature at the end point, the processor being operative to generate a curve between the start point and the end point, the curve being defined by a mathematical formula using the start point, the start tangent direction, the start curvature, the end point, the end tangent direction and the end curvature as variables.

87. The computer curve construction system of claim 86, and wherein the processor responds to additional signals received from the input device to determine an additional end point spaced from the end point, an additional end tangent direction at the additional end point and an additional end curvature at the additional end point, the processor being further operative to generate an additional curve between the end point and the additional end point, the additional curve being defined by the mathematical formula using the end point, the end tangent direction, the end curvature, the additional end point, the additional end tangent direction and the additional end point curvature as variables.

88. The computer curve construction system of claim 86, and wherein the curve comprises a cubic Bezier curve.

89. The computer curve construction system of claim 86, and further comprising:

a display device connected to the processor for displaying the curve.

90. A computer curve construction method for constructing curvature curves, the method comprising:

defining a start point;

defining a start tangent direction at the start point;

defining a start curvature at the start point;

defining an end point spaced from the start point;

defining an end tangent direction at the end point;

defining an end curvature at the end point; and

generating a curve between the start point and the end point, the curve being defined by a mathematical formula using the start point, the start tangent direction, the start curvature, the end point, the end tangent direction and the end curvature as variables.

91. The computer curve construction method of claim 90, and further comprising:

determining an additional end point spaced from the end point;

determining an additional end tangent direction at the additional end point;

determining an additional end curvature at the additional end point; and

generating an additional curve between the end point and the additional end point, the additional curve being defined by the mathematical formula using the end point, the end tangent direction, the end curvature, the additional end point, the addition end tangent direction and the additional end point curvature as variables.

92. The computer curve construction method of claim 90, and wherein the curve comprises a cubic Bezier curve.

93. The computer curve construction method of claim 90, and further comprising:

displaying the curve utilizing a display device.

94. A computer curve construction system for constructing curvature curves, the system comprising:

a input device; and

a processor coupled to the input device, wherein the processor responds to signals received from the input device to determine a start point, a start tangent direction at the start point, a start curvature at the start point, an end point spaced from the start point, a end tangent direction at the end point and an end curvature at the end point, the processor being operative to generate a curve between the start point and the end point, the curve being defined by mathematical formula using the start point, the start tangent direction, the start curvature, the end point, the end tangent direction and the end curvature as variables, and

wherein the start curvature is defined by a start curvature circle that includes the start point and has a center set along an imaginary line that includes the start point and is perpendicular to the start tangent direction, and

wherein the end curvature is defined by an end curvature circle that includes the end point and has a center set along an imaginary line that includes the end point and is perpendicular to the end tangent direction.

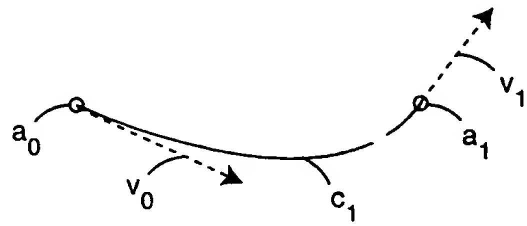



Fig.1
Prior Art

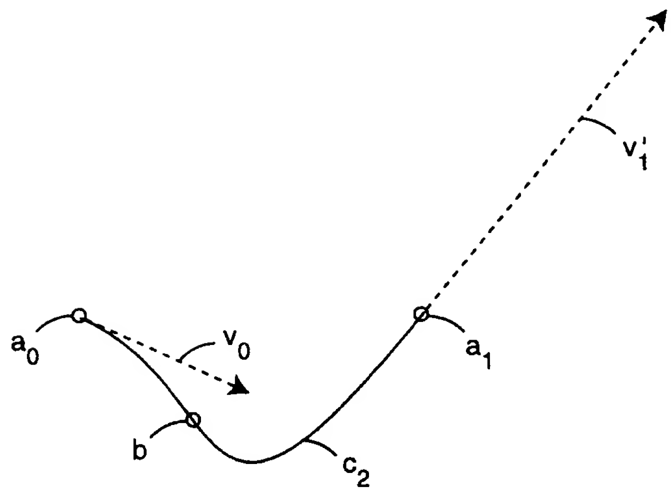



Fig. 2
Prior Art

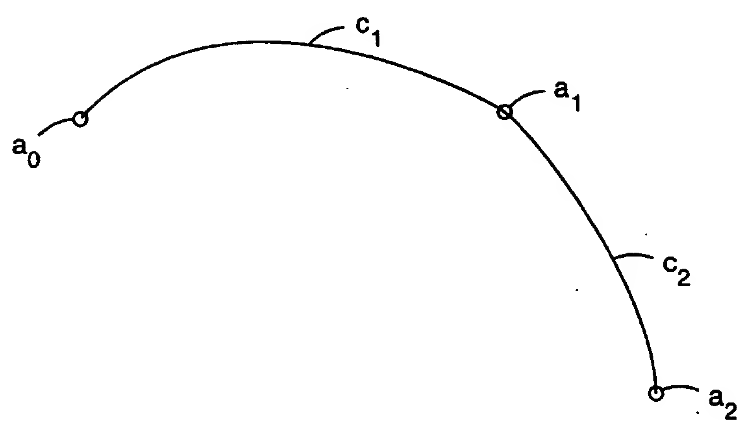



Fig.3
Prior Art

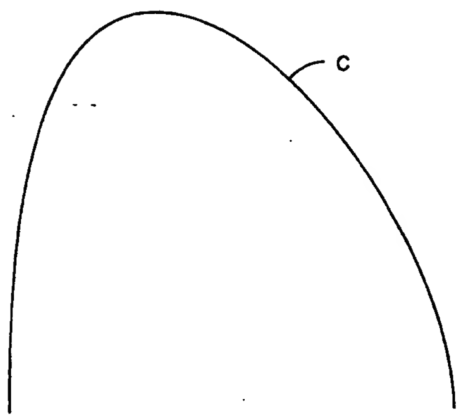



Fig.4
Prior Art

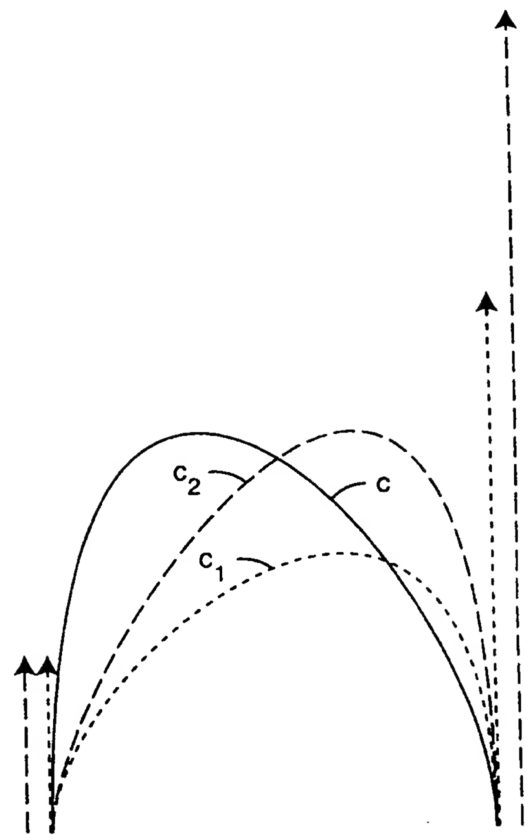



Fig.5
Prior Art

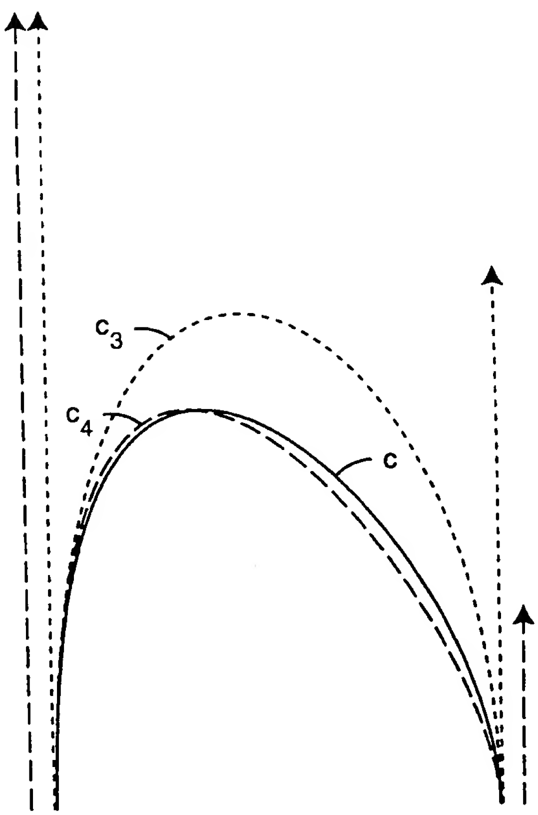



Fig.6
Prior Art

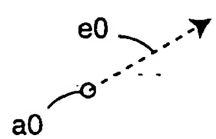



Fig.7

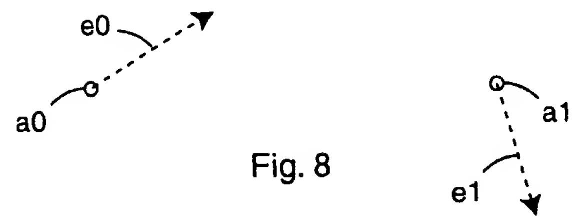



Fig. 8

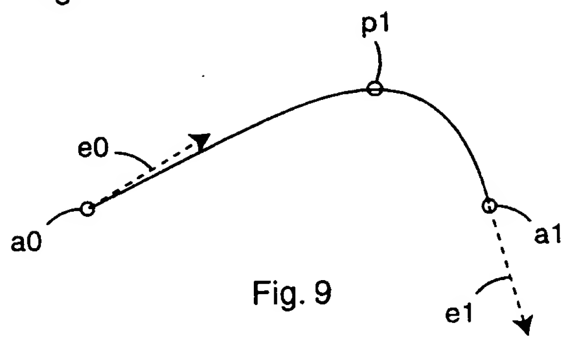



Fig. 9

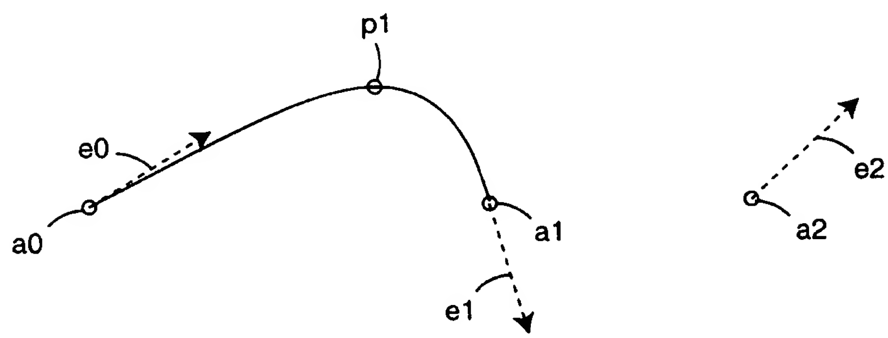



Fig.10

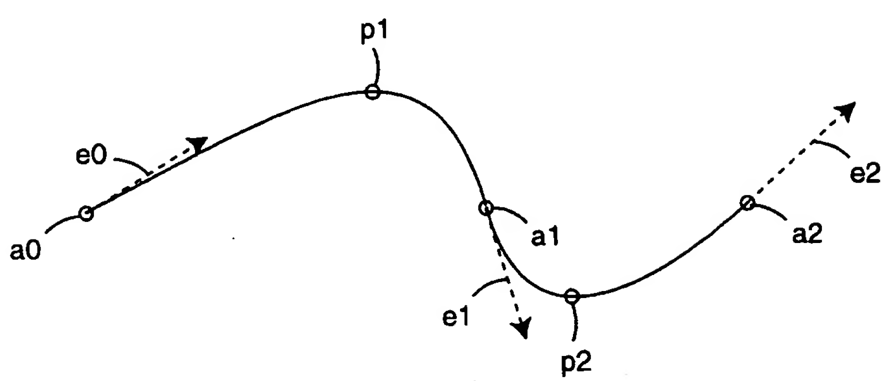



Fig.11

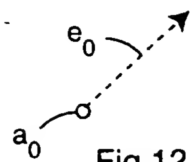



Fig.12

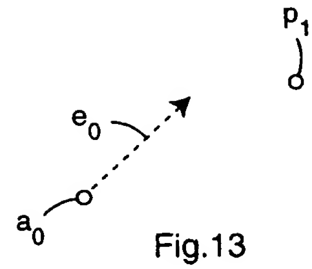



Fig.13

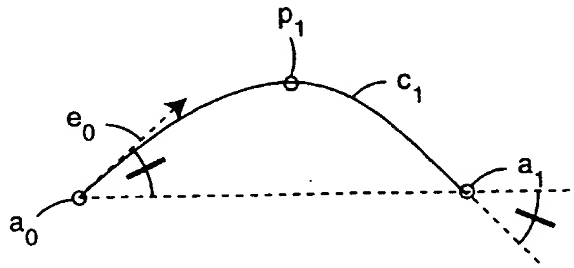



Fig.14

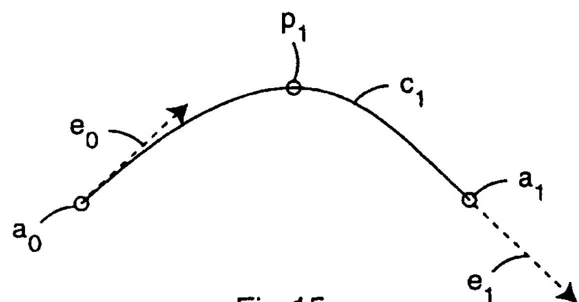



Fig.15

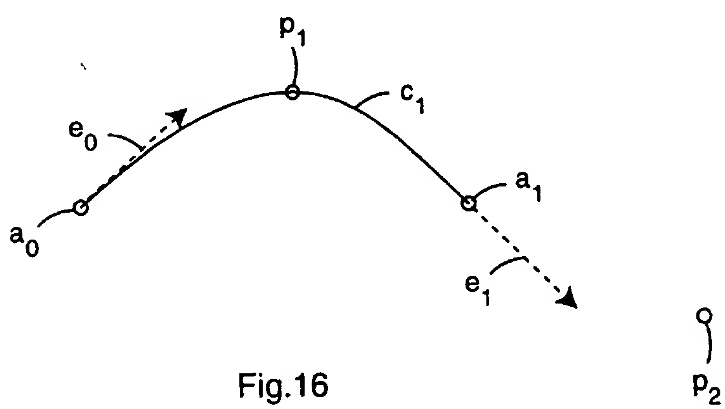



Fig.16

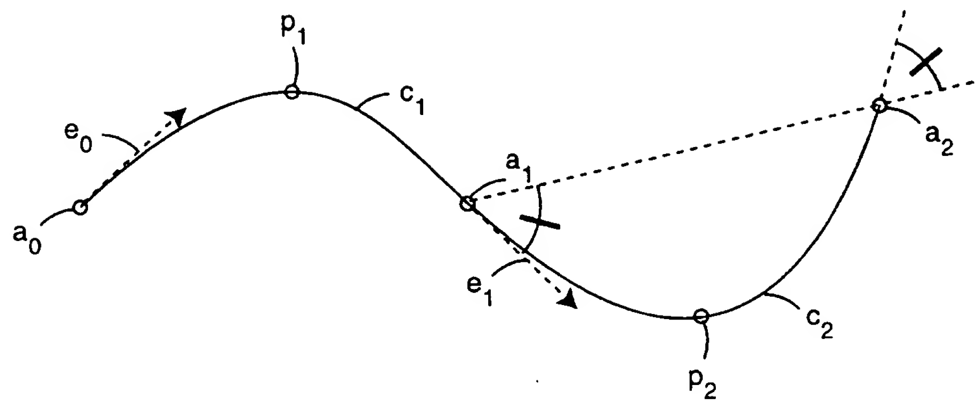



Fig.17

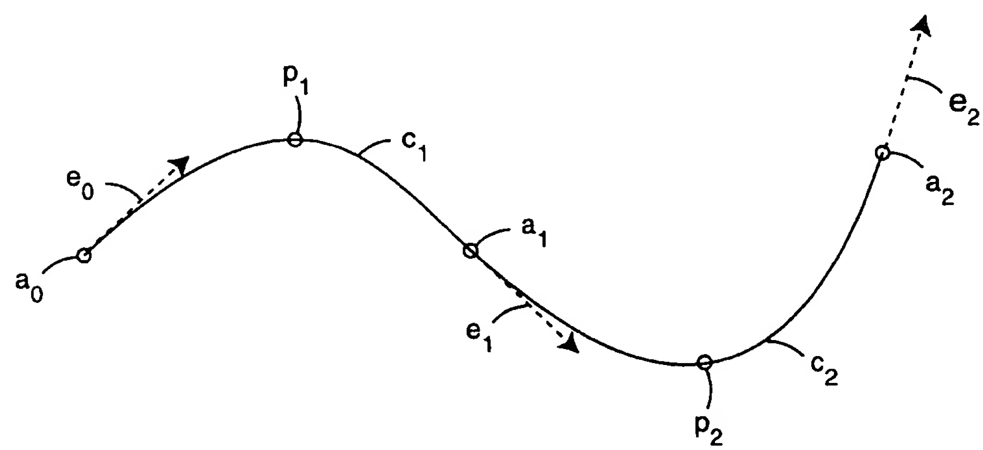



Fig.18

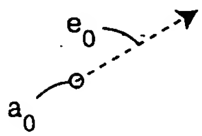



Fig.19

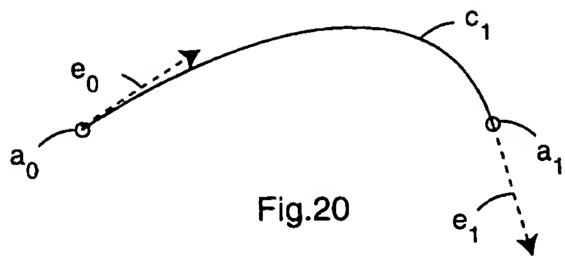



Fig.20

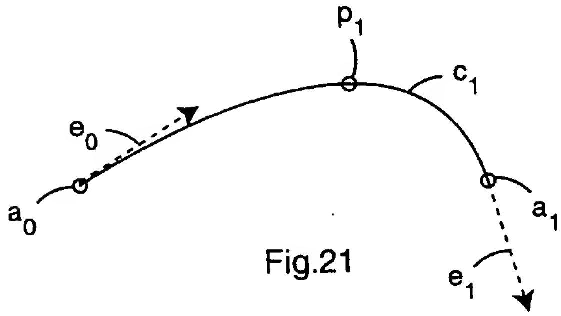



Fig.21

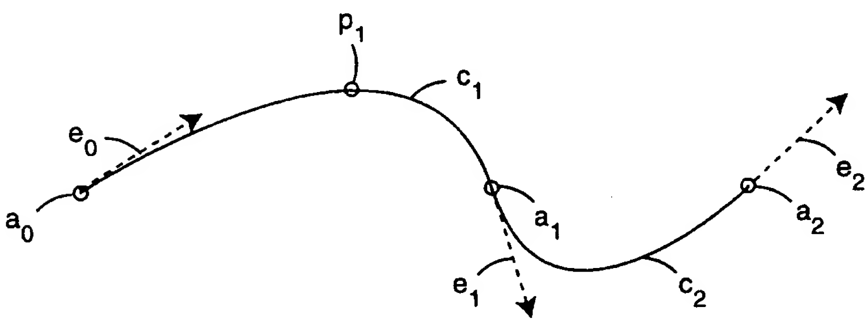



Fig.22

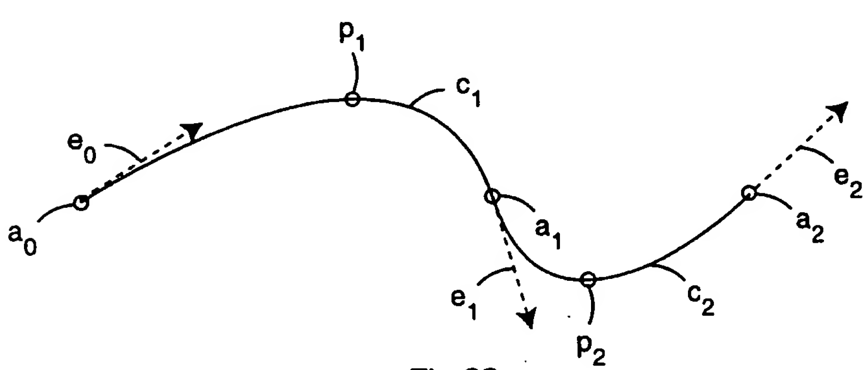



Fig.23

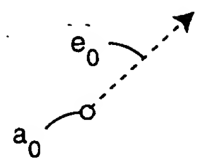



Fig.24

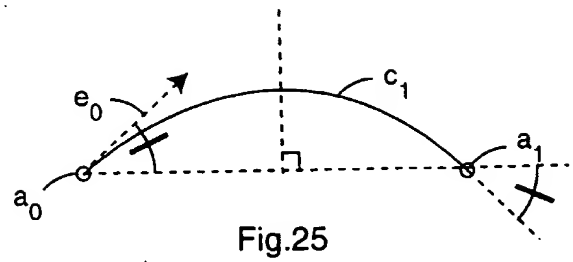



Fig.25

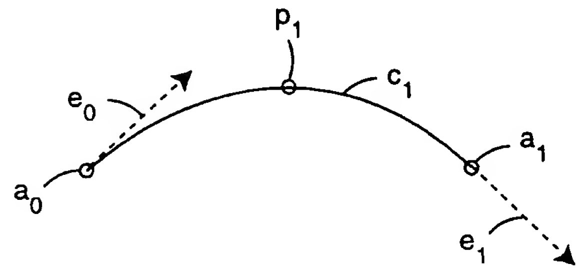



Fig.26

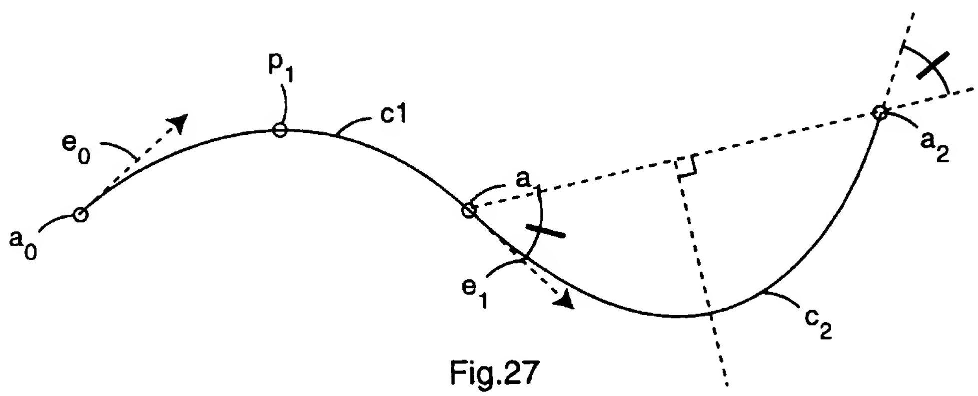



Fig.27

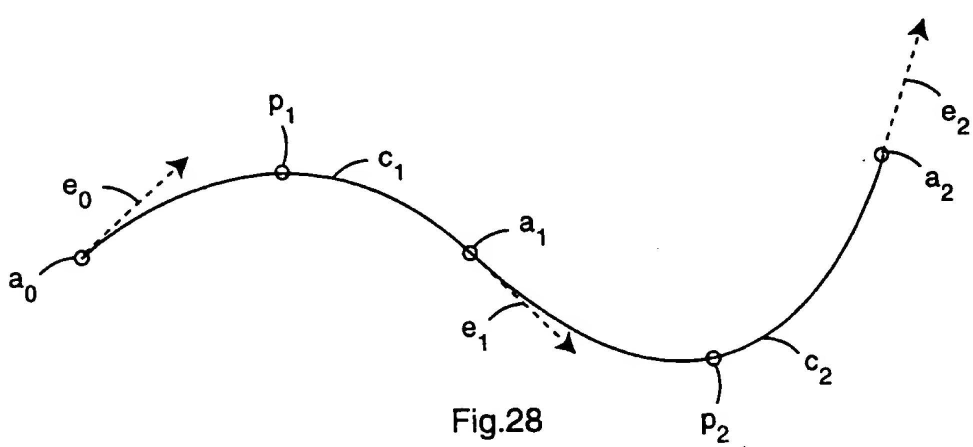



Fig.28

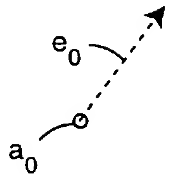



Fig. 29

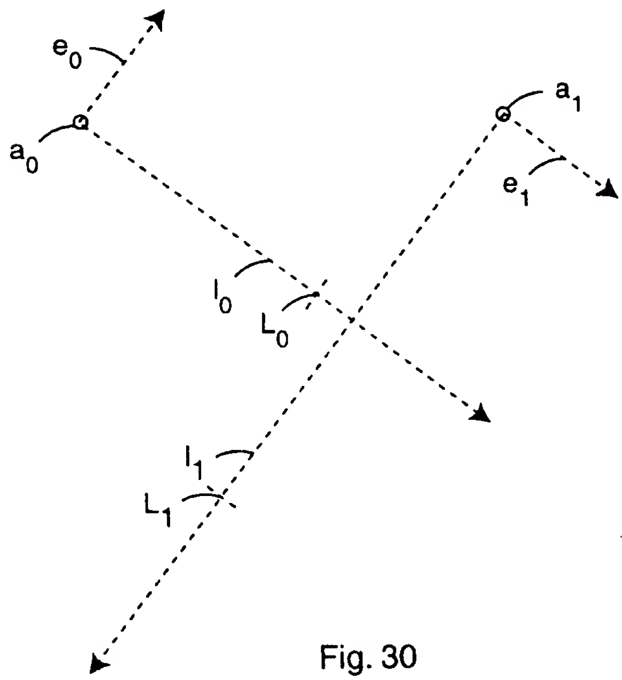



Fig. 30

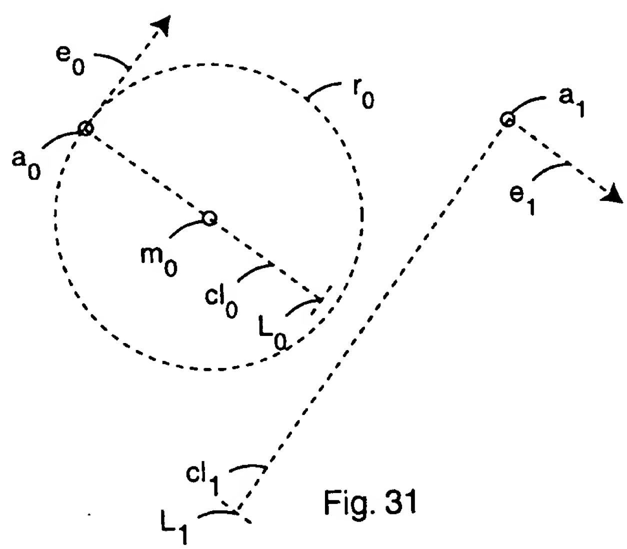
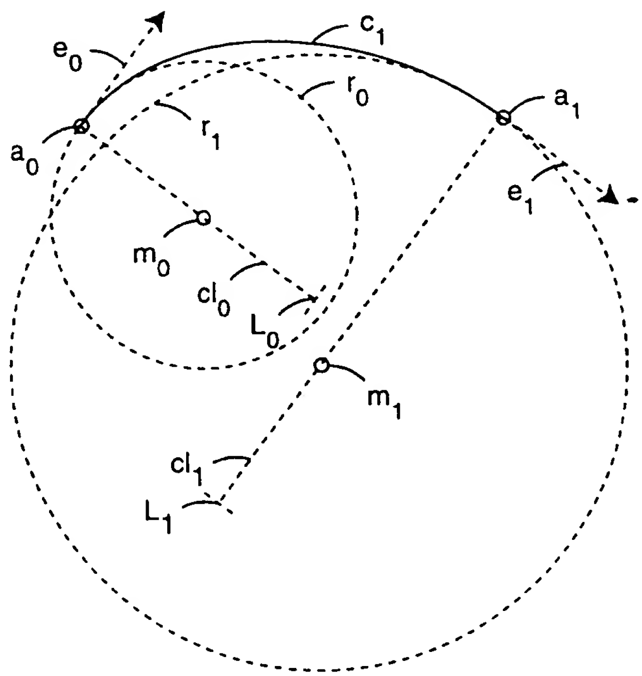
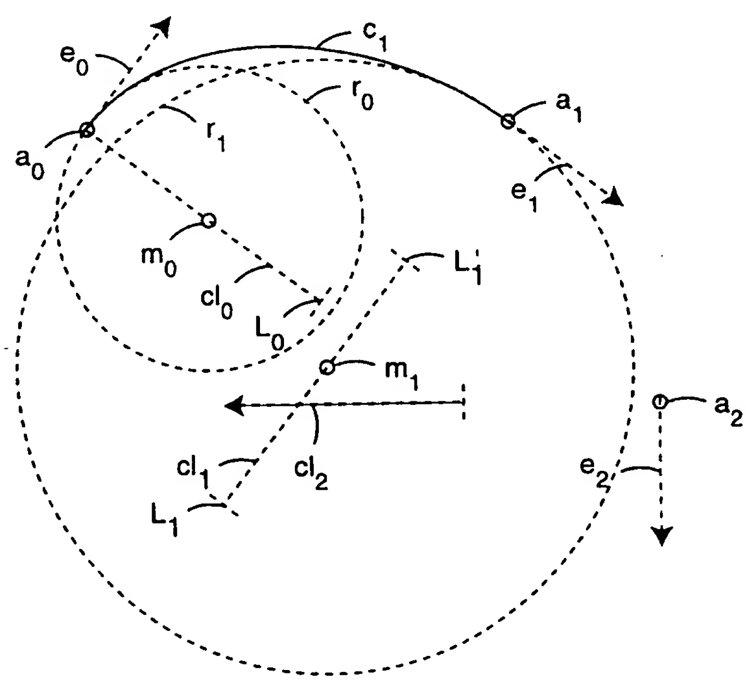



Fig. 31



e0
c1
r0
a1
r1
a0
e1
m0
cl0
L0
m1
cl1
L1


Fig.32



e0
c1
r0
a1
r1
a0
e1
m0
L1'
cl0
L0
m1
a2
cl1
cl2
e2
L1


Fig.33

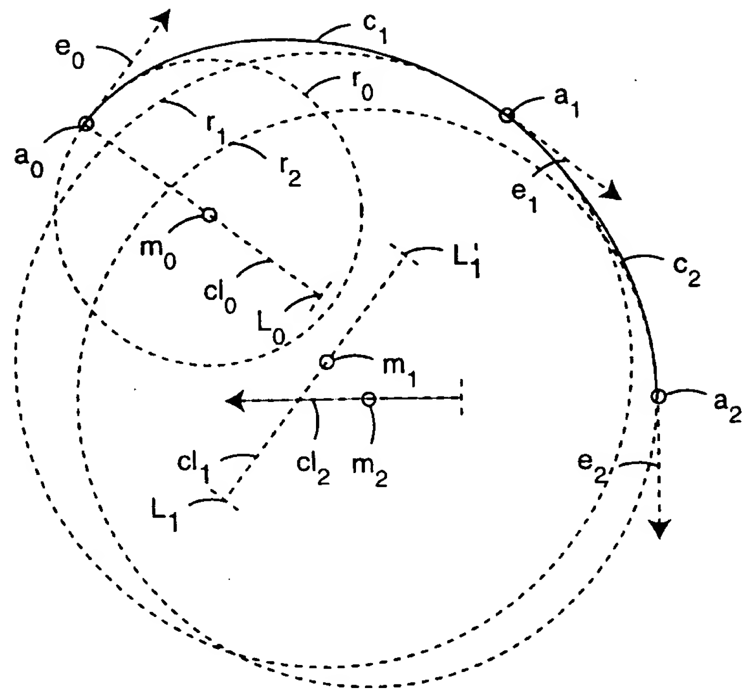

$c_1$
$e_0$
$r_0$
$a_1$
$r_1$
$a_0$
$r_2$
$e_1$
$m_0$
$L_1'$
$c_2$
$cl_0$
$L_0$
$m_1$
$a_2$
$cl_1$
$cl_2$
$m_2$
$e_2$
$L_1$

Fig.34

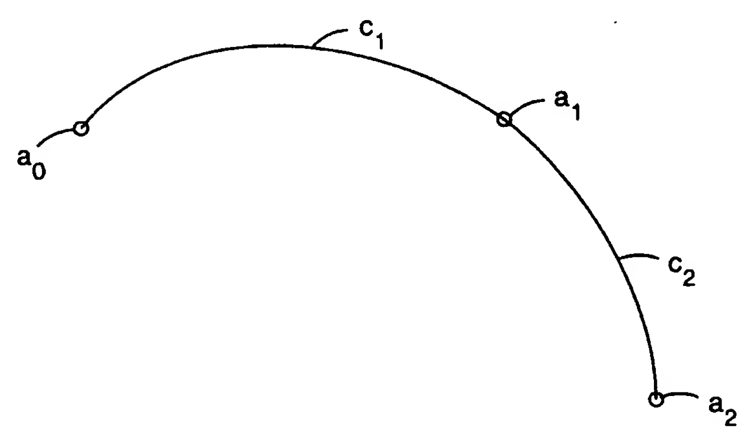

$c_1$
$a_1$
$a_0$
$c_2$
$a_2$

Fig.35

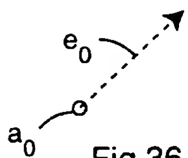



Fig.36

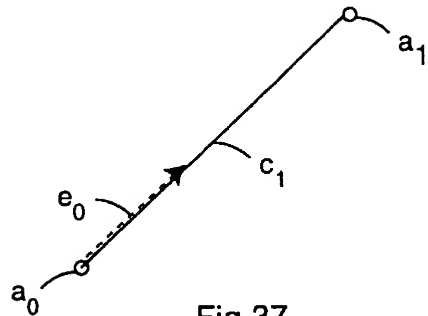



Fig.37

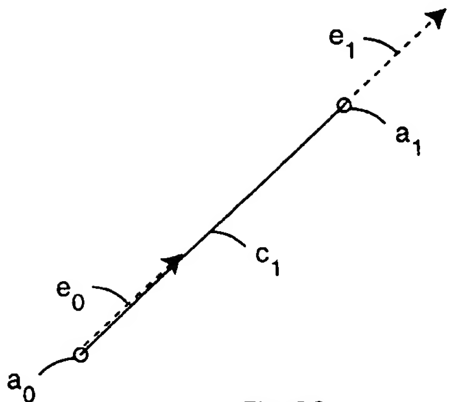



Fig.38

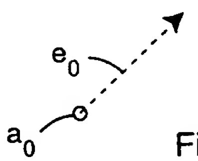



Fig.39

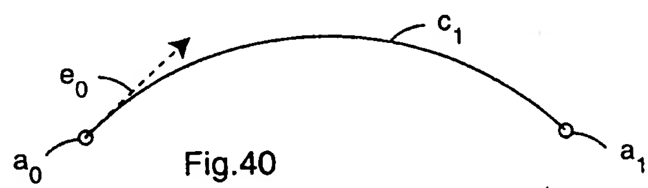



Fig.40

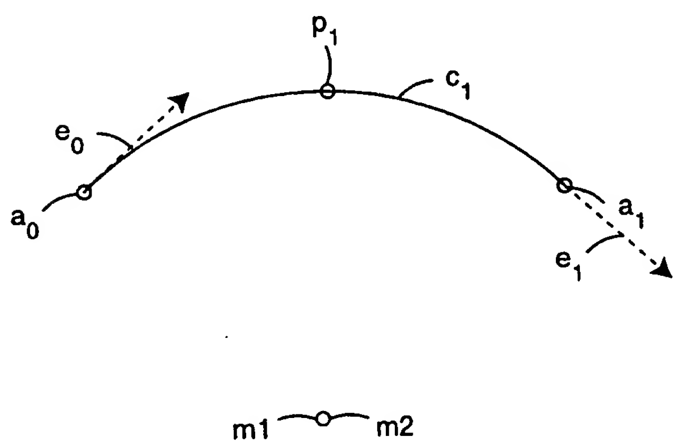



Fig.41

PATENT

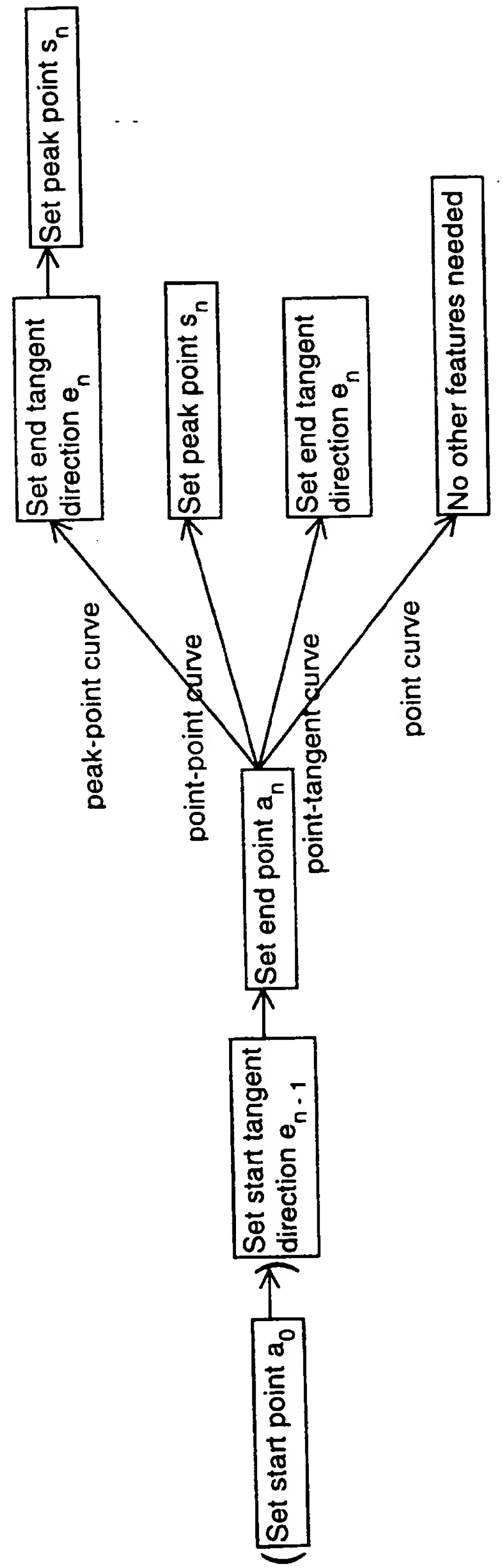



Fig.42

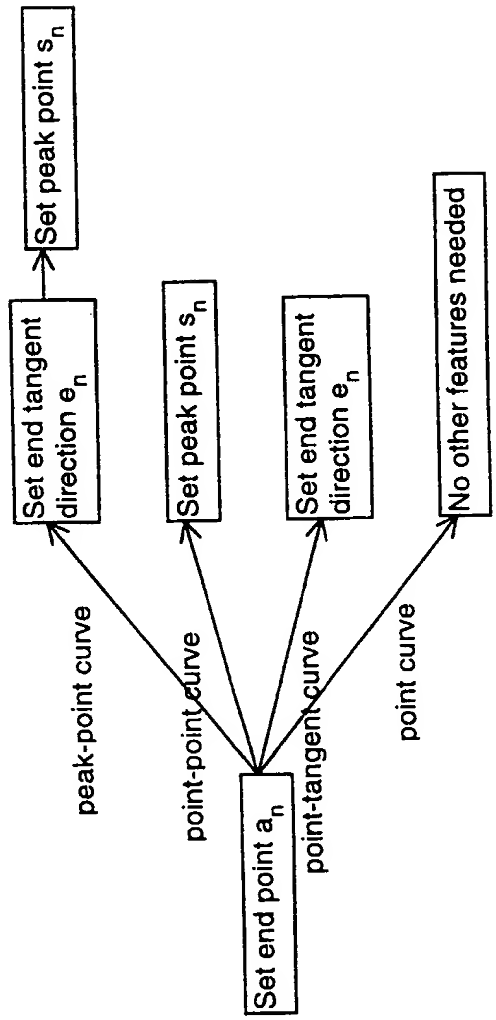



Fig.43

PATENT

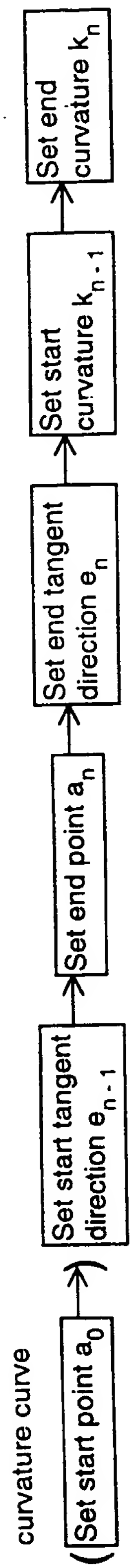



Fig.44

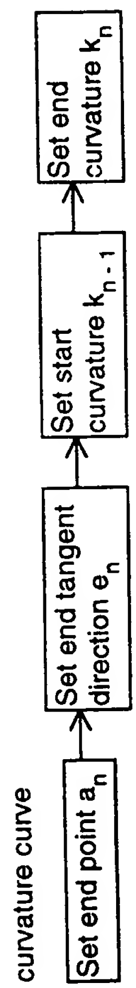



Fig.45

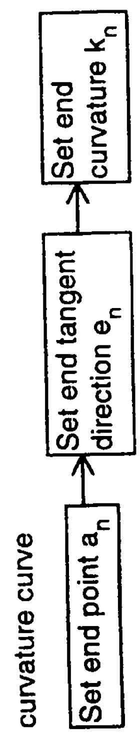



Fig.46

PATENT

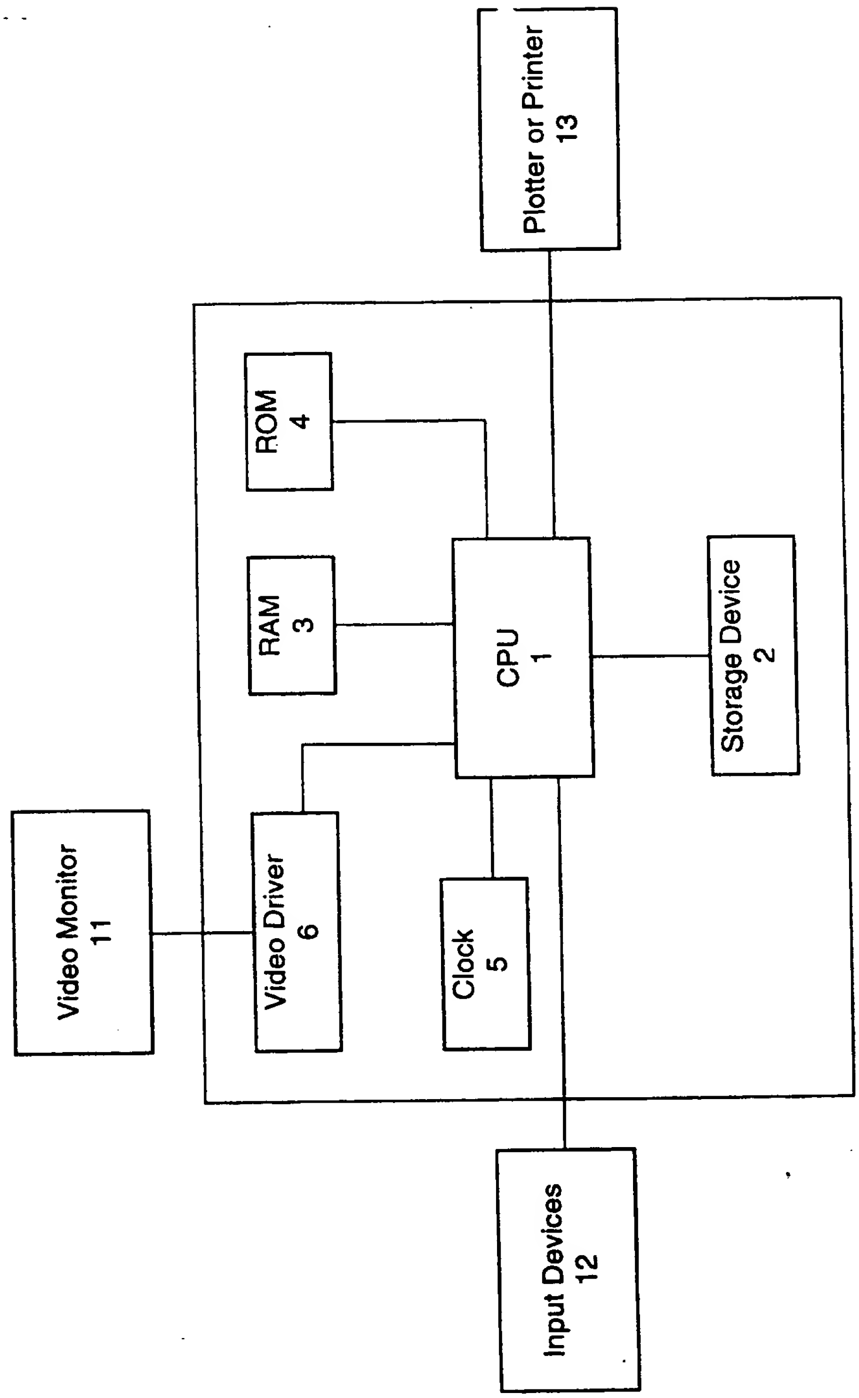



Fig.47

# INTERNATIONAL SEARCH REPORT

Inte   ional Application No
PCT/US 99/22285

**A. CLASSIFICATION OF SUBJECT MATTER**
IPC 7 G06T11/20

According to International Patent Classification (IPC) or to both national classification and IPC

**B. FIELDS SEARCHED**

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 G06T

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

**C. DOCUMENTS CONSIDERED TO BE RELEVANT**

| Category ° | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| X | PIEGL L: "INTERACTIVE DATA INTERPOLATION BY RATIONAL BEZIER CURVES" IEEE COMPUTER GRAPHICS AND APPLICATIONS,US,IEEE INC. NEW YORK, vol. 7, no. 4, April 1987 (1987-04), page 45-58 XP000038222 ISSN: 0272-1716<br>page 45, column 2, line 14 - line 18<br>page 48, column 1, line 6 - line 8<br>page 49, column 1, line 12 - line 37<br>page 52, column 1, line 34 - line 36<br>page 52, column 2, line 2 - line 6<br>page 53, column 2, line 9 - line 10<br>page 54, column 1, line 21 - line 24<br>page 55, column 2, line 9 - line 11<br>page 56, column 1, line 3 - line 6<br>figures 2,4<br>---<br>-/-- | 1-75 |

[X] Further documents are listed in the continuation of box C.

[ ] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

| Date of the actual completion of the international search | Date of mailing of the international search report |
|---|---|
| 24 January 2000 | 10/02/2000 |

| Name and mailing address of the ISA | Authorized officer |
|---|---|
| European Patent Office, P.B. 5818 Patentlaan 2<br>NL - 2280 HV Rijswijk<br>Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,<br>Fax: (+31-70) 340-3016 | González Arias, P |

5

Form PCT/ISA/210 (second sheet) (July 1992)

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

| Category ° | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| X | FORREST A R: "INTERACTIVE INTERPOLATION AND APPROXIMATION BY BEZIER POLYNOMIALS" COMPUTER AIDED DESIGN,GB,ELSEVIER PUBLISHERS BV., BARKING, vol. 22, no. 9, November 1990 (1990-11), page 527-537 XP000165205 ISSN: 0010-4485 page 533, line 1 - line 8 page 535, line 2 - line 3 figure 7 --- | 1-14, 56-67, 76-93 |
| X | HOSCHEK J: "CIRCULAR SPLINES" COMPUTER AIDED DESIGN,GB,ELSEVIER PUBLISHERS BV., BARKING, vol. 24, no. 11, November 1992 (1992-11), page 611-618 XP000328796 ISSN: 0010-4485 the whole document --- | 68-71 |
| X | POTTMANN H: "LOCALLY CONTROLLABLE CONIC SPLINES WITH CURVATURE CONTINUITY" ACM TRANSACTIONS ON GRAPHICS,US,ASSOCIATION FOR COMPUTING MACHINERY, NEW YORK, vol. 10, no. 4, October 1991 (1991-10), page 366-377 XP000248361 ISSN: 0730-0301 page 367, line 10 - line 16 figure 2 ----- | 82-93 |

5